UNIVERSIDAD AUTÓNOMA DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería de Tecnologías y servicios de Telecomunicación

TRABAJO FIN DE GRADO

Reconocimiento del locutor dependiente del texto: Experimentos con la base de datos RSR2015

Álvaro Mesa Castellanos
Tutor: Doroteo Torre Toledano

Mayo 2016

Reconocimiento del locutor dependiente del texto: Experimentos con la base de datos RSR2015

AUTOR: Álvaro Mesa Castellanos
TUTOR: Doroteo Torre Toledano

Área de Tratamiento de Voz y Señales (ATVS)
Dpto. Tecnología Electrónica y de las Comunicaciones
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Mayo de 2016

Resumen (castellano)

Este Trabajo Fin de Grado tiene como objetivo el estudio y desarrollo de un sistema de reconocimiento de locutor dependiente de texto, empleando para ello modelos ocultos de Markov (HMMs), ampliamente usados en reconocimiento de voz. Para ello, se partirá de resultados y pruebas realizados anteriormente en el grupo ATVS en esta temática, se intentará replicar experimentos previos, extenderlos a la nueva base de datos RSR2015 y, en la medida de lo posible, mejorar las prestaciones de los sistemas previos.

En primer lugar, emplearemos dos bases de datos para el desarrollo del sistema y realización de experimentos, YOHO y RSR2015. La primera de ellas constituye la base de datos de referencia para el reconocimiento de locutor dependiente de texto, ampliamente utilizada en los últimos años en el ámbito de la investigación, y que ha sido objeto de estudio y evaluaciones durante los últimos años. La segunda de ellas será sobre la que realizaremos la mayor parte de los experimentos, al tratarse de una base de datos de reciente aparición. Partiremos de los resultados obtenidos por el grupo ATVS en reconocimiento de locutor dependiente de texto con la base de datos YOHO, los cuales se replicarán y servirán de punto de partida para realizar los experimentos con la nueva base de datos RSR2015. Con ella se realizarán distintas pruebas en cuanto a verificación de locutor y verificación de frase se refiere.

Por lo tanto, se buscará obtener resultados similares a los ya existentes para la base de datos RSR2015 y se intentará introducir nuevas mejoras para alcanzar mejores resultados.

Abstract (English)

The aim of this Bachelor Thesis is the study and development of a text-dependent speaker recognition system, using hidden Markov models as they are widely used in speech recognition. For that purpose, results and tests previously done by ATVS group will be taken, we will try to replicate and extend them to the new RSR2015 database and to the maximum extent possible, improve the performance of previous systems.

First of all, we will use two databases for the developing of the systems and conducting of experiments, YOHO and RSR2015. The first one has been one of the reference databases for the text-dependent speaker recognition task, widely used in the last years in research and which has been under study and evaluations. We will focus on the second one to conduct the experiments as it is a recently appeared database. We will start from results obtained by ATVS group in text-dependent speaker recognition task with the YOHO database, which will be replicated and will be a starting point to conduct the new experiments with the new RSR2015 database. Different tests as for speaker and sentence verification will be done with this database.

Overall, we will try to get similar results to the ones that already exist for the RSR2015 database, and we will try to make further improvements to achieve better results.

Palabras clave (castellano)

Reconocimiento de locutor dependiente de texto, Modelos ocultos de Markov, Adaptación al locutor, Verificación de locutor, RSR2015, YOHO, Kaldi, HTK.

Keywords (English)

Text-dependent speaker recognition, Hidden Markov Models, Speaker adaptation, Speaker verification, RSR2015, YOHO, Kaldi ,HTK.

Agradecimientos

En primer lugar, me gustaría dar las gracias a mi tutor Doroteo Torre por la preocupación de mi aprendizaje y comprensión del tema que estabámos tratando, así como sus continuos ánimos y consejos para seguir adelante. También darle las gracias por haberme ofrecido la posibilidad de realizar este trabajo.

Agradecer a mis amigos de toda la vida y a mis amigos del Lago por hacerme más llevaderos todos estos años de carrera, y por todo el tiempo que hemos vivido juntos. Agradezco también a mis compañeros de clase por todos los momentos que hemos vivido durante toda la carrera.

Agradecer a mi familia, en especial a mis padres, su continuo apoyo, tanto en lo personal como en lo académico, durante todos estos años, creyendo siempre en mí y apoyándome en mis decisiones, sin ellos esto habría sido muy difícil. Agradezco especialmente a mi hermano mayor, Sergio, por todo lo que me ha enseñado en la vida. Por último, agradecer a Patri por su apoyo durante este año y por todo.

INDICE DE CONTENIDOS

INDICE DE FIGURAS

INDICE DE TABLAS

1 Introducción

1.1 Motivación

La señal de voz constituye un rasgo biométrico muy adecuado para la autentificación remota debido principalmente a su amplia extensión, es decir, las comunicaciones y canales de voz son muy frecuentes en la sociedad actual debido a la generalización de dispositivos móviles y uso de ordenadores, además de la ya implementada red telefónica tradicional. Es por ello que la voz se presenta como un elemento fácilmente accesible ya que no se necesita otro dispositivo de adquisición o transmisión adicional. Por lo tanto, este hecho confiere a la voz una gran ventaja sobre otros rasgos biométricos, en cuanto a autentificación remota se refiere.

Además de proporcionarnos información sobre la identidad del locutor, la voz incorpora múltiples fuentes de información, tales como el idioma, edad y sexo, emociones, registro, estilo y origen geográfico, además del contenido semántico del mensaje.

Evidenciada la riqueza de información contenida en la señal de voz, ésta supone un rasgo biométrico proclive a la realización de sistemas de verificación y autenticación remota de locutor.

Debido al creciente uso de Internet y los dispositivos móviles para acceder a distintas cuentas, tales como cuentas bancarias o datos privados almacenados en Internet, es evidente la necesidad de autenticar al usuario y dotarle de acceso a los distintos sistemas en cualquier lugar, momento y dispositivo. En Internet tradicionalmente se emplean contraseñas que el usuario escribe desde su ordenador, éstas se cifran y se transmiten a través del ordenador de éste. En otros sistemas de acceso, cuentas bancarias o acceso a un recinto protegido se emplea un pin, exclusivo del usuario y conocido sólo por éste. Sin embargo, la principal vulnerabilidad de estos sistemas reside en la posible filtración de contraseñas y acceso a éstas por parte de otros usuarios. Es por ello que la autentificación biométrica ha cobrado gran importancia en los últimos años. La señal de voz, entre otros rasgos biométricos, nos da la posibilidad de autenticarnos ante un sistema de una manera más segura, ya que ésta contiene información única y exclusiva para los distintos locutores [2].

Podemos encontrar tres grandes tipos de aplicaciones que se benefician de la información biométrica presente en la señal de voz:

- Autentificación por voz: conocido como verificación del locutor o *natural voice checking*, que consiste en emplear la voz como mecanismo de identificación
- Detección de locutor: conocido como *speaker" spotting*, que consiste en detectar la presencia de un locutor en un flujo de audios (por ejemplo de llamadas).
- Reconocimiento de locutores en ámbitos forenses: que implica el uso de la voz como prueba en procedimientos legales.

En este trabajo nos centraremos en la primera de estas aplicaciones. En particular, nos centraremos en el siguiente escenario de aplicación: dado un pin conocido por un usuario, el sistema de autenticación por voz debe determinar si la locución dicha por el locutor coincide con el pin y además si la voz es del usuario que dice ser o no.

La principal motivación es que el grupo ATVS ya tenía un sistema desarrollado y probado con una base de datos antigua. Con la reciente aparición de la base de datos RSR2015, que es el nuevo estándar para la evaluación de sistemas de autenticación por voz, se pretende actualizar los resultados de la tecnología ya desarrollada y, en la medida de lo posible, mejorarla.

1.2 Objetivos

El principal objetivo es el de evaluar el sistema de autenticación por voz desarrollado por el grupo ATVS sobre la nueva base de datos RSR2015, para de ese modo evaluar el estado de la tecnología sobre una tarea mucho más compleja que la tarea sobre la que se había probado antes, y en la medida de lo posible mejorar la tecnología en función de los resultados que se obtengan sobre esa base de datos.

Por ello, los objetivos propuestos son, tomando como punto de partida el trabajo realizado anteriormente en el grupo ATVS:

1. Replicar los resultados obtenidos con la base de datos YOHO haciendo uso de la herramienta HTK (describiremos esta herramienta en el capítulo de Diseño).
2. Partiendo de esos resultados, crear un sistema de reconocimiento de locutor dependiente de texto con la nueva base de datos RSR2015, haciendo uso también de la herramienta HTK, para compararlos con la referencia de los resultados obtenidos en el paper [3].
3. Aunque no estaba inicialmente previsto, en el desarrollo del TFG se ha conseguido un tercer objetivo: desarrollar un nuevo sistema de autenticación por voz haciendo uso de la nueva herramienta de reconocimiento de voz Kaldi (también descrita en el apartado de Diseño) y empleando unos modelos de reconocimiento de voz en inglés muchos más avanzados que los usados anteriormente.

1.3 Organización de la memoria

La memoria consta de los siguientes capítulos:

- **Capítulo 1: Introducción**
  Contiene una breve introducción a la temática abordada en el trabajo, así como la motivación para realizar el trabajo y los objetivos propuestos.
- **Capítulo 2: Estado del arte**
  Contiene una descripción de las tecnologías empleadas para el reconocimiento de locutor. Se abordará la problemática de la verificación de locutor en modo text-dependent, así como el reconocimiento de voz y adaptación al locutor.
- **Capítulo 3: Diseño**
  En este capitulo describiremos las herramientas software utilizadas para llevar a cabo este trabajo, así como las bases de datos empleadas.
- **Capítulo 4: Desarrollo**
  En este capítulo se explicará cómo se ha desarrollado este trabajo y llevado a cabo los experimentos de verificación de locutor. Se describirán los procedimientos de verificación de locutor y de frase, y cómo los hemos implementado con las herramientas software de las que se disponen. Además, se explicarán los modelos empleados en los experimentos, así como las fases de entrenamiento y test para ambas bases de datos, YOHO y RSR2015.

- **Capítulo 5: Integración, pruebas y resultados**
  En este apartado se mostrarán los resultados obtenidos de los experimentos con la base de datos YOHO y RSR2015. Para la primera de ellas, el experimento de verificación de locutor y comparación con los resultados previamente obtenidos y para la segunda, los experimentos de verificación de locutor y frase, comparándolos con los resultados ya existentes.
- **Capítulo 6: Conclusiones y trabajo futuro**
  En este capítulo se expondrán las conclusiones obtenidas derivadas de la realización de este trabajo así como las propuestas de trabajo futuro de éste.

2 Estado del arte

2.1 Reconocimiento de locutor

El objetivo del reconocimiento de locutor es determinar la identidad del locutor en un segmento de voz. La forma de determinar la identidad se puede dar de tres maneras dependiendo de la aplicación:

- Verificación o Autenticación: El locutor reclama una identidad y el sistema analiza la voz para comprobar si el locutor es quien dice ser o no (clasificación binaria). Es el objeto de estudio de este trabajo.
- Identificación: consiste en decidir a qué locutor pertenece la voz dentro de un conjunto de locutores conocidos (clasificación multiclase). Se puede dar en un conjunto abierto o cerrado.
- Detección: consiste en detectar un locutor concreto en un flujo de grabaciones amplio (por ejemplo, de llamadas telefónicas). La decisión es también binaria (ausencia o presencia del locutor) pero al contrario que en la verificación, el locutor no realiza ninguna acción cooperativa con el sistema.

Dentro del campo de reconocimiento de locutor tenemos dos grandes grupos en función de la tecnología empleada, el reconocimiento independiente del texto y el reconocimiento dependiente del texto.

2.1.1 Dependiente del texto

En los sistemas dependientes del texto o "text-dependent", el locutor coopera con el sistema reclamando una identidad y diciendo una frase clave específica, una palabra clave o una secuencia de dígitos, por ejemplo, y el sistema verifica que la locución pertenece a la identidad reclamada y que además lo dicho coincide con el texto requerido. Este tipo de sistemas son ampliamente utilizados en control de acceso biométrico y en concreto en la aplicación de autentificación por voz. El texto requerido por el sistema puede variar de un acceso a otro, con el fin de evitar el acceso a impostores, los cuales podrían grabar al locutor diciendo la contraseña o frase concreta [2]. En este trabajo nos vamos a centrar en la autenticación por voz, y por tanto en la verificación de locutor dependiente del texto.

2.1.2 Independiente del texto

El reconocimiento de locutor independiente de texto es la técnica que se emplea en las aplicaciones de detección de locutores y de detección forense de locutores. En este caso, el locutor no coopera con el sistema de ningún modo, por lo que no se conoce lo que el éste dice. El objetivo es reconocer al locutor independientemente de lo dicho. En este trabajo no se va a trabajar en este tipo de técnicas, por lo que no se profundizará más en su estudio.

2.2 Funcionamiento de un sistema de verificación de locutor

La tarea de verificación de locutor consiste en, dado un audio de entrada al sistema y una identidad reclamada, verificar que el locutor es realmente quien dice ser y no otra persona, y además, verificar que lo dicho por esta persona coincide con un pin o una frase requerida por el sistema.

La configuración típica para un sistema de verificación automática de locutor genérica es:

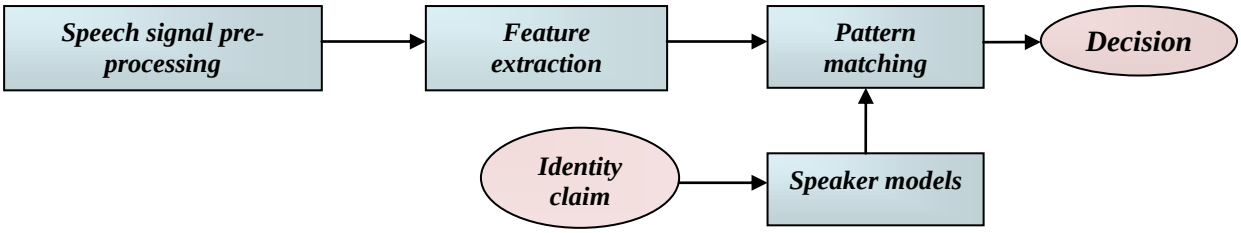

Figura 2-1: Esquema genérico de un sistema de verificación automática de locutor [1]

La entrada al sistema es una muestra de voz perteneciente a un locutor desconocido, pero que pretende autenticarse con una identidad reclamada. En primer lugar se lleva a cabo un pre-procesado de señal. A continuación, se lleva a cabo un proceso de extracción de características de la muestra de voz, típicamente basado en MFCCs los cuales explicaremos más adelante. A partir de la identidad reclamada por el locutor, el sistema toma un modelo de locutor determinado, entrenado previamente con los datos pertenecientes al locutor real cuya identidad se reclama. Finalmente, el bloque "pattern matching" se encarga de comparar las características del audio del locutor desconocido con el modelo de la voz de la identidad reclamada, generando la decisión binaria de autenticar o no al locutor.

Habitualmente la comparación de características consta de dos fases:

- En la primera de ellas se compara la muestra de entrada, *X*, respecto a un modelo universal de voz (Universal Background Model, UBM). En nuestro caso denotaremos este modelo universal como λ_{SI} y será entrenado con una gran cantidad de locuciones pertenecientes a un gran número de locutores. De este modo obtenemos un modelo universal independiente del locutor (*speaker independent, si*) y de la comparación de la voz con este modelo una puntuación o "score_{si}", definida como la probabilidad:

$$score_{si} = P(X|\lambda_{si})$$

- En la segunda de ellas, se compara la muestra de entrada, X, respecto al modelo de locutor correspondiente a la identidad reclamada. En nuestro caso este modelo será un modelo dependiente del locutor (speaker dependent, sd) que denotaremos como λ_{sd}. Para obtener este modelo se parte del UBM, que se adapta con una pequeña cantidad de locuciones del locutor en cuestión. Como la cantidad de voz que se puede solicitar a un locutor para crear su modelo en una situación práctica es pequeña, no es posible crear un modelo nuevo desde cero con esta voz, y se recurre a técnicas de adaptación al locutor de modelos idependientes del locutor. De esta forma obtenemos una puntuación o "score_{sd}", definida como la probabilidad:

$$score_{sd} = P(X|\lambda_{sd})$$

Si *X* es la locución de entrada y ϑ un umbral determinado a fijar, el sistema determina si esa locución pertenece o no a un locutor cuya identidad se reclama de acuerdo con el criterio de máxima verosimilitud (Maximum Likelihood, ML). Se obtiene así el resultado del cociente entre dos verosimilitudes:

$$\frac{P(X|\lambda_{sd})}{P(X|\lambda_{si})} \quad \begin{cases} \geq \vartheta\,, aceptar\ locutor \\ < \vartheta\,, rechazar\ locutor \end{cases}$$

ϑ es el umbral de decisión, que dependerá de los resultados obtenidos para cada experimento y el cual debemos fijar con el fin de controlar la relación entre las probabilidades de aceptar y rechazar a un locutor, y así ajustar el compromiso entre los errores en ambos sentidos posibles de la decisión.

Habitualmente se emplea el logaritmo de ese cociente y resulta:

$$\Lambda(X) = \log\left(P(X|\lambda_{sd})\right) - \log\left(P(X|\lambda_{si})\right).$$

En cualquier sistema biométrico se cometerán errores de decisión de dos tipos:

- *Falsa aceptación (FA):* si decidimos que un fragmento de voz pertenece a la identidad reclamada pero en realidad pertenece a un locutor distinto.
- *Falso rechazo (FR):* si decidimos que un fragmento de voz no pertenece a la identidad reclamada cuando sí pertenecía realmente a dicha identidad reclamada.

El valor de estas dos medidas determinará el comportamiento del sistema, esto es, cómo éste es capaz de discriminar entre el locutor real que reclama su identidad, y que llamaremos target, y un impostor que reclama una identidad que no es la suya, y que llamaremos non-target. Por lo tanto, definiremos la discriminación como la habilidad del sistema de separar entre scores target y non-target [5].

A continuación, mostramos dos distribuciones típicas de puntuaciones target (azules) y non-target (rojas) a la izquierda, y sus respectivas curvas de FA y FR en función del umbral:

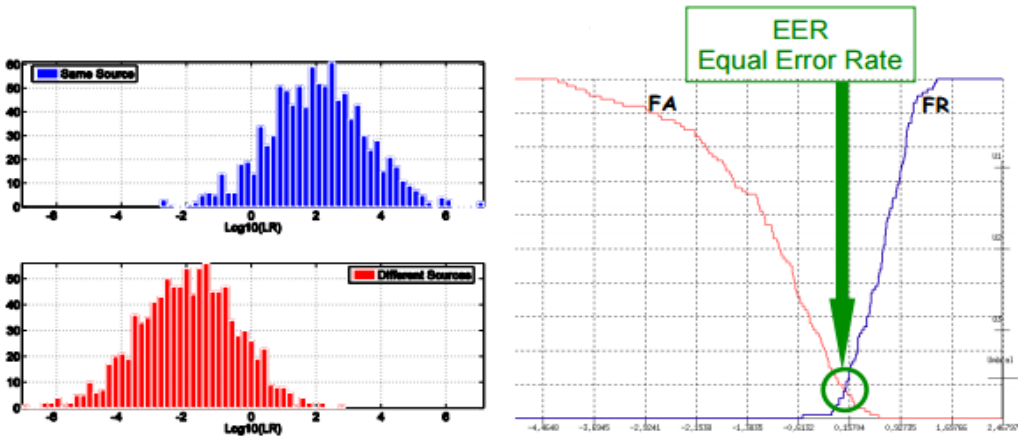

Figura 2-2: Distribución de puntuaciones y medida del error de decisión [5]

En la figura de la derecha el eje de abscisas representa el valor del umbral, así que recorriendo el eje tenemos un valor de FA y FR determinado. Por lo tanto, el valor del umbral lo escogeremos en función de las necesidades concretas de aplicación en cuestión y de lo restrictivos que queramos ser en cuanto al FR y FA. El punto en el que la FA es igual al FR se denota como Tasa de Igual Error o Equal Error Rate (EER).

2.3 Modelado de la voz y reconocimiento de voz

El reconocimiento de voz es la tecnología que trata de obtener una transcripción textual a partir de una grabación de voz. Este trabajo no se centra en el reconocimiento de la voz, pero va a hacer uso extensivo de tecnología de reconocimiento de voz, ya que la tecnología de reconocimiento de voz permite modelar de forma muy detallada tanto el contenido léxico de la voz (la transcripción) como las características particulares de la voz de cada locutor (mediante técnicas de adaptación al locutor). Ambas posibilidades nos van a permitir realizar la verificación del locutor dependiente de texto con tecnología muy similar a la del reconocimiento de voz. Por ello vamos a describir brevemente los mecanismos que se emplean en reconocimiento de voz para modelar la voz y para realizar la adaptación al locutor.

Dado que el reconocimiento de voz tiene como objetivo obtener la transcripción textual de contenidos de voz, la medida más natural de la tasa de error en este problema es la tasa de error de palabra o Word Error Rate (WER), y se define como:

$$WER = \frac{S+B+I}{N} \text{x} 100\ \%$$

Donde S es el número de sustituciones, B el número de borrados, I el número de inserciones de palabras encontradas al comparar la frase reconocida con la de referencia, y N es el número de palabras de la frase de referencia.

2.3.1 Modelo de lenguaje, modelo léxico y modelo acústico-fonético

A fin de modelar el habla y ser capaces de extraer la transcripción textual de una grabación de voz, un reconocedor de voz maneja varios modelos del habla en distintos niveles.
De mayor a menor nivel de abstracción, el primer modelo a tener en cuenta es el modelo de lenguaje. Un modelo de lenguaje tiene como objetivo predecir la palabra que seguirá a una secuencia de palabras dada. Para este efecto se hace uso de un modelado estadístico basado en *n-gramas* que modela la probabilidad de aparición de secuencias de *n* palabras. El modelo de lenguaje es una pieza clave para reducir la tasa de error de palabra.

El siguiente modelo a tener en cuenta es el léxico que emplea nuestro reconocedor. Éste está formado por todas las palabras que el sistema es capaz de reconocer, asociando a cada una de éstas su transcripción fonética correspondiente.

El último elemento, sin duda el más complejo, es el modelo acústico-fonético. Un modelo acústico-fonético tiene como objetivo modelar cómo suenan los distintos fonemas (alófonos) de una lengua. En nuestro caso este modelado se hace a partir de modelos ocultos de Markov, (Hidden Markov Models, HMMs) fonéticos, en el que cada fonema aparece modelado como un HMM y las palabras se forman concatenando los HMMs asociados a los fonemas que las componen. Los HMMs no modelan directamente la señal de voz, sino una secuencia de vectores de características extraídos típicamente cada 10 ms, a partir de la señal de voz, en una etapa que se denomina de extracción de características y que es bastante estándar (con algunas variaciones menores) en todo tipo de procesado de voz (incluyendo el reconocimiento del locutor en todas sus modalidades).

2.3.2 Extracción de características

Para la creación de modelos fonéticos de la voz es necesario extraer características acústicas de ella. Las características acústicas más habituales son los llamados MFFCs

(coeficientes mel-frequency cepstrum). Estos coeficientes responden al uso de la escala Mel, como escala ajustada en frecuencia a las características conocidas de la percepción auditiva humana.

El proceso de parametrización de la voz queda descrito por las siguientes etapas [5]:

1. ***Pre-énfasis:*** cuyo objetivo es compensar la atenuación en altas frecuencias, provocada por la impedancia de radiación.
2. ***Extracción de tramas***: En nuestro caso, por ejemplo, se extraen tramas de 25 ms, tomadas cada 10 ms.
3. ***Enventanado:*** se multiplica en el tiempo la trama por una ventana, que en nuestro caso será Hamming.
4. ***Cálculo de los parámetros MFCC:***
    a. ***Cálculo de la DFT:*** se aplica la DFT sobre una señal enventanada y se toma el módulo, descartando la fase.
    b. ***Banco de filtros de escala MEL:*** aplicamos al módulo de la DFT un banco de filtros de escala perceptual MEL, que emula las bandas críticas del oído humano. En la siguiente figura vemos un ejemplo de la distribución del banco de filtros hasta 5KHz:

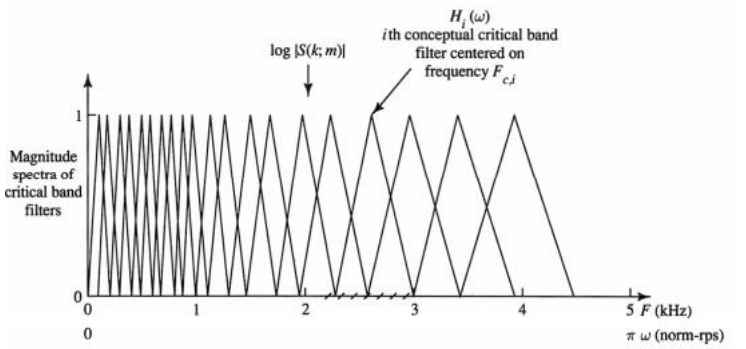

Figura 2-3: Banco de filtros [5]

    c. ***Cálculo de la energía en cada banda:*** la salida del banco de filtros anterior está relacionada con la energía en cada banda. Se calcula la energía total logarítmica en cada una de las bandas críticas.
    d. ***Aplicación de la transformada cepstral:*** sobre la salida del proceso anterior se aplica una transformación final de tipo DCT, quedándonos con los primeros coeficientes (típicamente 13).

2.3.3 Modelos ocultos de Markov (HMMs)

Los modelos ocultos de Markov o HMMs son una técnica de modelado estadístico que ha venido utilizándose en reconocimiento de voz durante las últimas décadas. Modelan estadísticamente la acústica de la voz, esto es, como suenan los distintos fonemas y palabras. Un HMM es una máquina de estados finita, en la que las observaciones son una función probabilística del estado, es por esto que se le considera un proceso doblemente estocástico y que está formado por el HMM en sí, que modela el proceso de producción de voz teniendo en cuenta la variabilidad temporal, y por un proceso observable, el cual tiene en cuenta la variabilidad espectral de la voz y las características de ésta.

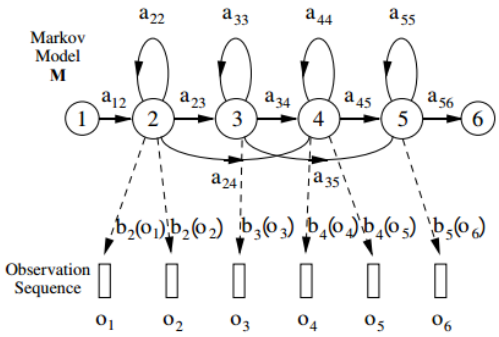

Figura 2-4: Cadena de Markov de 5 estados [6]

En la figura podemos observar un ejemplo de cadena de Markov [6] en la que los estados 1 y 5 no generan ninguna salida (en la implementación que se realiza en HTK). Como vemos, en reconocimiento de voz, típicamente se emplean cadenas de Markov que van de izquierda a derecha y están formadas por tres estados posibles (representando la parte inicial, media y final del fonema).

Una cadena de Markov está formada por los siguientes elementos [7]:

- *N*: número de estados del modelo, en nuestro caso de estudio *N* será 3 para los modelos de fonemas. Los modelos de voz son modelos "de izquierda a derecha" o de Bakis.
- *M*: número de las distintas observaciones posibles en cada estado. En nuestro caso de reconocimiento de voz las observaciones tendrán un carácter continuo por lo que este número no está definido (es infinito).
- Distribución de probabilidad de las transiciones entre estados. Indica la probabilidad de saltar de un estado a otro. Más concretamente es la probabilidad de estar en un tiempo *t* en el estado S_i condicionado a estar en un estado S_j en un tiempo *t+1*. La distribución viene denotada por la matriz:

$$A = \{a_{ij}\}, \quad \text{donde} \quad a_{ij} = P[q_{t+1} = S_j | q_t = S_i] \quad 1 \leq i, j \leq N$$

- La función de distribución de probabilidad de observación de símbolo en un estado *j*, denotada por la matriz:

$$B = \{b_j(k)\}, \quad \text{donde} \quad b_j(k) = P[v_k \; at \; t \; | q_t = S_j] \qquad \begin{matrix} 1 \leq j \leq N \\ 1 \leq k \leq M \end{matrix}$$

  En el caso del reconocimiento de voz las observaciones son los vectores de características continuos, por lo que en lugar de funciones distribución de probabilidad se manejarán funciones densidad de probabilidad multidimensionales (cada una de ellas de tantas dimensiones como el número de componentes que tengan los vectores de características). Estas funciones densidad de probabilidad se modelarán paramétricamente como mezclas de Gaussianas multidimensionales.

- La distribución inicial de estados $\pi = \{\pi_i\}$, donde,

$$\pi_i = P[q_1 = S_i], \quad 1 \leq i \leq N.$$

En nuestro caso sólo habrá un estado inicial con probabilidad 1 (el primer estado del primer fonema, que normalmente modelará el silencio inicial antes de la pronunciación), siendo el resto de probabilidades 0.

De esta manera, el modelo queda determinado por:

$$\lambda = (A, B, \pi).$$

Por lo tanto, un HMM nos permite modelar estadísticamente la acústica de la voz. La utilidad de los HMMs en el procesamiento de voz está relacionada con las soluciones de los siguientes problemas conocidos de los HMMs:

- ***Cálculo de la probabilidad de una secuencia de observaciones dado el modelo, P(O|λ).*** Se calcula a partir de unas variables denominadas "forward" y "backward", gracias a las cuales es posible calcular esa probabilidad buscada. Esta probabilidad nos permitirá comparar la probabilidad de varios modelos para la misma observación. Si disponemos de modelos para las distintas palabras que puede reconocer un reconocedor, esto nos permitiría determinar cuál es la palabra que explica mejor la observación.
- ***Cálculo de la secuencia de estados que mejor explica una secuencia de observación dada.*** Se calcula a partir del algoritmo de Viterbi que nos proporciona el camino de máxima verosimilitud para la secuencia observada. Si estamos trabajando con un modelo HMM complejo con múltiples caminos que se correspondan con distintas pronunciaciones, el algoritmo de Viterbi nos permitirá obtener el camino (i.e. la secuencia de fonemas) reconocida. Existen también extensiones que permiten buscar no sólo el mejor camino sino los N mejores caminos, algo que resulta útil cuando se tienen datos adicionales sobre los resultados del reconocimiento o cuando se desea hacer una estimación del grado de confianza en el reconocimiento.
- ***Cálculo de los parámetros del modelo para maximizar P(O|λ).*** Se realiza a partir del algoritmo de Baum-Welch (Expectation Maximization, EM), que garantiza, para cada iteración:

$$P(O|\lambda') \geq P(O|\lambda) \text{ , Siendo } \lambda\text{' el modelo reestimado.}$$

Este algoritmo es el que se usa en el entrenamiento de los modelos. Empleando un conjunto de secuencias de voz con sus correspondientes transcripciones ortográficas es posible reestimar los parámetros del modelo de modo que el modelo explique mejor los datos observados.

Es conveniente observar que el método de aprendizaje no es discriminativo sino generativo: sólo tratamos de modelar de la forma más precisa los datos de entrenamiento, pero los modelos no tratan de mejorar la discriminación entre los distintos fonemas (por ejemplo). Esa es una de las grandes ventajas de otros modelos que están empleándose cada vez más en reconocimiento de voz: las redes neuronales artificiales, y en particular las redes neuronales profundas (Deep Neural Networks, DNNs).

2.3.4 Adaptación al locutor

En los sistemas de reconocimiento de locutor partimos de un modelo que denominamos modelo de background universal (Universal Background Model, UBM), el cual modela la acústica de la voz y es entrenado con un gran número de horas de voz pertenecientes a muchos locutores distintos.

En nuestro caso, reconocimiento de locutor dependiente de texto, no entrenamos un modelo del locutor de magnitud parecida al UBM para cada locutor, ya que eso implicaría tomar una gran cantidad de voz para cada locutor. Evidentemente, esto reportaría mejores resultados, al disponer de más datos de entrenamiento, pero resulta inviable en un sistema práctico, en donde dispondremos solamente de unas pocas locuciones del usuario. Para solucionar este problema surgen las llamadas técnicas de adaptación al locutor, cuyo objetivo es, partiendo del UBM, adaptar este modelo con el fin de modelar lo más fielmente posible la acústica de la voz de un locutor en concreto.

Las distintas técnicas de adaptación al locutor que analizaremos son, *MLLR*, *fMLLR* y *MAP*.

2.3.4.1 Adaptación MLLR y fMLLR

La adaptación *MLLR* (Maximum Likelihood Linear Regression) [6] es una técnica de adaptación de modelos que estima un conjunto de transformaciones lineales para la media y varianza de las distintas mezclas de gaussianas (GMM) de un modelo HMM. El objetivo es que la probabilidad de que cada estado del HMM inicial genere las locuciones de adaptación sea mayor. Las matrices de transformación son obtenidas a partir de la técnica *EM* (Expectation-Maximization).

Las medias adaptadas se calculan como:

$$\mu' = W\varepsilon$$

Donde W es la matriz de transformación y ε el vector de medias extendido (con un uno para conseguir una transformación afín en vez de lineal). A su vez, W se descompone en dos matrices,$W = [b\ A]$, donde A es una matriz de transformación NxN y b un vector de bias.

La adaptación *fMLLR*, también conocida como "feature-space*" MLLR (fMLLR)* [8], es una transformación afín de las características, de la forma:

$$x \rightarrow Ax + b \quad \Leftrightarrow \quad x \rightarrow Wx^{+}$$

$$x^{+} = \begin{bmatrix} x \\ 1 \end{bmatrix}$$

Tanto *fMLLR* como *MLLR* permiten adaptar con menos datos, además, ambas técnicas permiten almacenar solo la matriz de transformación. Por estos motivos, en nuestros experimentos emplearemos la adaptación *MLLR* o *fMLLR*, al disponer de pocos datos de entrenamiento.

2.3.4.2 Adaptación MAP

Denominada también adaptación Bayesiana [9], es una técnica de adaptación que opera parámetro a parámetro de los HMM y considera que el parámetro del modelo independiente del locutor es la información a priori sobre dicho parámetro.

Con la adaptación *MAP* debemos fijar un factor de adaptación, que indica el peso que le damos a la información nueva.

Sin embargo, al operar parámetro a parámetro hace que se requieran muchos más datos de adaptación [9], es por ello que en nuestras pruebas descartamos el uso de *MAP*.

2.3.4.3 Clases de regresión

Las clases de regresion aumentan la flexibilidad en el proceso de adaptación del locutor. Si disponemos de pocos datos de entrenamiento se aplicará una transformación global a todas las medias y varianzas de las gaussianas. Por el contrario, si disponemos de bastantes datos de adaptación es posible establecer una serie de clases de regresión, cuyo objetivo es modificar los parámetros de un grupo determinado de gaussianas, separando éstas en clases y tratándolas de distinta forma. Como resultado, se establecerán transformaciones para un grupo específico de gaussianas, pudiendo agrupar estas, por ejemplo, por el tipo de fonema.

Cuando aplicamos *MLLR* o *fMLLR* se determina cuántos datos de adaptación hay disponibles para cada clase del árbol de regresión, determinando para cada nodo si hay o no suficientes datos. Por ello, solo se entrenan transformaciones para clases con suficientes datos y que tienen hijos sin suficientes datos. En la siguiente figura vemos un ejemplo de ello, donde los nodos 5,6 y 7 carecen de suficientes datos de adaptación.

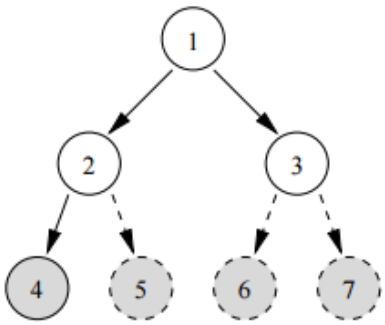

Figura 2-5: Árbol de regresión [9]

3 Diseño

3.1 Herramientas utilizadas: HTK

El "Hidden Markov Model Toolkit" (HTK) [6] es una herramienta que construye y manipula modelos ocultos de Markov. HTK es usado esencialmente en el ámbito de la investigación de reconocimiento de voz aunque se ha usado en otras aplicaciones tales como reconocimiento de caracteres y síntesis de voz.

La herramienta consiste en un conjunto de módulos y librerías escritas en código C y proporciona utilidades para el análisis de voz, entrenamiento de HMMs, testeo y análisis de resultados. La herramienta se desarrolló en los años 90 y actualmente su licencia pertenece a Microsoft.

La arquitectura es la siguiente:

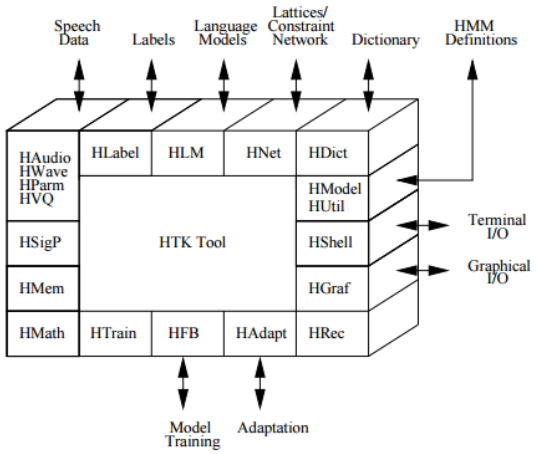

Figura 3-1: Componentes de HTK [6]

Las herramientas utilizadas en el proyecto serán explicadas en la fase de desarrollo de este trabajo.

3.2 Herramientas utilizadas: Kaldi

Kaldi es una herramienta para reconocimiento de voz escrita en C++ [8] que tiene la licencia Apache License v2.0 y es ampliamente utilizada, principalmente en entornos de investigación de reconocimiento de voz.

Kaldi es similar en cierto modo a la herramienta previamente descrita HTK, sin embargo, su principal objetivo es tener un código más moderno (se empezó a utilizar hacia 2010) y flexible (a ella se incorporan rápidamente las últimas novedades de investigación en reconocimiento de voz) que ésta. Sus principales características son las siguientes:

- Integración a nivel de código con transductores de estados finitos (o FSTs), empleado éstos como una librería.
- Soporte para operaciones de álgebra lineal, donde se incluye una librería de matrices basada en el estándar BLAS y las rutinas LAPACK.
- Opta por un diseño extensible ya que se proporcionan los algoritmos de la manera más genérica posible, con el fin de facilitar el desarrollo de sistemas.

- Proporciona scripts que realizan tareas comunes en el ámbito de reconocimiento de voz, como por ejemplo la creación de un modelo de lenguaje o adaptación al locutor. Estos scripts serán tomados como base a la hora de realizar los distintos experimentos, realizando las modificaciones oportunas. Además, está diseñado y pensado para ser ejecutado en máquinas de muchos núcleos y grandes prestaciones.

3.3 Bases de datos

Para la realización de este trabajo se han utilizado dos bases de datos, YOHO y RSR2015. La primera de ellas ha sido utilizada para replicar los experimentos ya realizados por el grupo ATVS y la segunda para realizar nuevos experimentos de reconocimiento de locutor dependiente de texto. Por otro lado, para el entrenamiento de los modelos de reconocimiento de voz se han empleado las bases de datos TIMIT y Switchboard.

3.3.1 YOHO

La base de datos YOHO surge con el principal objetivo de establecer un marco de comparación entre sistemas de verificación de locutor y fomentar la investigación en este aspecto. La base de datos tiene las siguientes características [10]:

- Contiene frases de pares de dígitos (12-34-56).
- Está formada por 138 locutores, 108 hombres y 30 mujeres.
- Grabada en un entorno real de oficina, en condiciones controladas y de poco ruido.
- Se compone de datos de entrenamiento y verificación. En el entrenamiento tenemos 4 sesiones de 24 frases por sesión. Los datos de verificación se adquirieron en 10 sesiones espaciadas una media de 3 días entre ellas, con 4 frases por sesión.
- Está muestreada a 8KHz con un ancho de banda analógico de 3.8KHz.

En este escenario dependiente de texto, se muestra un texto (text-prompted) conocido por el sistema y se le pide al usuario que lo diga. La sintaxis empleada consiste en secuencias de tres pares de dígitos, a modo de contraseña ("54-22-62" → "fifty-four, twenty-two, sixty-two").

3.3.2 RSR2015

3.3.2.1 Visión general

La base de datos RSR2015 se creó para el desarrollo, entrenamiento y testeo de sistemas de verificación de locutor dependientes de texto. El objetivo [11], es el de proporcionar a la comunidad científica una base de datos que permita distintos tipos de protocolos, es por ello que contiene locuciones de diferentes longitudes, frases cortas y secuencias aleatorias de números.

El porqué del desarrollo de otra base de datos se fundamenta en los siguientes puntos: [11]

- Proporcionar una base de datos de un tamaño similar a aquellas destinadas a tratar con reconocimiento de locutor independiente de texto.
- Tener una base de datos equilibrada en cuanto a locutores femeninos y masculinos se refiere.
- Permitir el análisis de la variabilidad fonética en un contexto de reconocimiento dependiente de texto.

La base de datos está formada por 300 locutores, 157 hombres y 143 mujeres, y por cada locutor tenemos 3 sesiones de entrenamiento de 73 locuciones cada una, y 6 sesiones de verificación con 73 locuciones cada una, lo que da un total de 657 locuciones en 9 sesiones por locutor.

Las locuciones fueron grabadas por el "Institute for Infocomm Research" (I2R), en Singapore, usando los siguientes dispositivos:

- Samsung Nexus.
- Samsung Galaxy S.
- Samsung Tab.
- HTC Desire.

Cada locutor fue grabado al menos usando 3 dispositivos diferentes.

La tasa de muestreo es de 16KHz, la cual tuvimos que modificar a 8KHz para poder operar con los modelos que disponíamos. El audio está codificado con 16 bits.

En cuanto a la distribución de los locutores según su origen y género, tenemos lo siguiente:

	Female	***Male***	***Total***
Chinese	118	119	79%
Malay	14	28	14%
Others	11	10	7%

Tabla 3-1: Distribución de locutores en RSR2015 [11]

3.3.2.2 Organización de la base de datos

La base de datos está formada por 4 grupos de frases bien diferenciados. El primer grupo consiste en frases cortas para una verificación de locutor por frase. Contiene 30 frases en inglés tomadas de TIMIT, comunes a todas las sesiones y locutores. La media de la duración de las 30 frases de los 300 locutores es de 3.2 segundos. Contiene 71 horas de voz. Un ejemplo de ello:

"030: the redcoats ran like rabbits"

La segunda parte consiste en 30 comandos cortos diseñados para hogares inteligentes controlados por voz común a todas las sesiones y locutores. Contiene 45 horas de voz. Un ejemplo de ello:

"038: Call brother"

La tercera y última parte, que será objeto de nuestro estudio, consiste en 3 secuencias de 10 dígitos en un orden aleatorio y 10 secuencias de 5 dígitos en orden aleatorio también, además, las secuencias de dígitos son dependientes de la sesión. Contiene en 35 horas de voz. Un ejemplo de ello:

"08,061: 7-2-9-8-0-1-6-4-3-5"

En los experimentos realizados se hace uso de la tercera parte de la base de datos, para el caso de secuencias de dígitos aleatorias de 10 números. En este caso el sistema conoce una secuencia aleatoria de 10 dígitos y lo muestra (text-prompted). Este texto debe coincidir con lo dicho por el locutor, de lo contrario se rechazará su acceso. Los ficheros de entrenamiento y test se explicarán con detalle en la siguiente sección.

Por último, los locutores se presentan en dos grupos, "development" y "evaluation".

	Male	***Female***
Development	M051 → M100	F048 → F094
Evaluation	M101 → M157	F095 → F143

Tabla 3-2: Distribución de locutores por subcorpus y género en RSR2015 [11]

3.3.3 Switchboard (SWBD)

La base de datos Switchboard (SWBD), ha sido utilizada para entrenar los modelos acústico-fonéticos de reconocimiento de voz empleados en las pruebas realizadas con Kaldi. El entrenamiento ha sido realizado por Alicia Lozano, doctorando del grupo ATVS, por lo que en este TFG se han tomado los modelos acústico-fonéticos completos ya entrenados.

Esta base de datos tiene las siguientes características:

- Consiste en audios recogidos de habla telefónica espontánea en dos canales.
- Los datos están muestreados a 8KHz.
- La cantidad de voz está en torno a 300 horas de voz.

3.3.4 TIMIT

La base de datos TIMIT [12] se usa del mismo modo que la Switchboard, para entrenar los modelos acústicos con los que empezar a trabajar en HTK. El entrenamiento de estos modelos fue realizado por el grupo ATVS. La base de datos TIMIT tiene las siguientes características:

- Diseñada para proporcionar habla para el desarrollo y evaluación de sistemas de reconocimiento de habla automáticos.
- Se trata de voz microfónica en inglés americano.
- El corpus lo forman 630 locutores. Entre ellos hay 438 hombres (70%) y 192 mujeres (30%).
- Cada locutor lee un total de 10 frases, divididas en entrenamiento y test.
- El contenido total de voz está en torno a 5.4 horas.

4 Desarrollo

4.1 Marco General

Las dos bases de datos empleadas para la realización de los experimentos han sido YOHO y RSR2015. Con ellas realizaremos experimentos relacionados con la verificación de locutor y con la verificación de la frase dicha por éste.

4.2 YOHO

4.2.1 Descripción general y modelos empleados

Para llevar a cabo los experimentos con la base de datos YOHO se ha empleado la herramienta HTK. En la configuración general para implementar el sistema de reconocimiento de locutor dependiente de texto con HTK se emplean una serie de ficheros y programas que describiremos a continuación.

Para la creación de los vectores de características (MFCCs) se hace uso de la herramienta HCopy a la que hay que pasarle un fichero de configuración que, en nuestro caso, tiene las siguientes características:

- Tamaño de ventana: 25 ms,
- Desplazamiento de ventana: 10 ms,
- Banco de filtros de 26 canales,
- Salida de 12 coeficientes MFCCs más la energía normalizada.

En los experimentos empleamos clases de regresión para agrupar y modificar las gaussianas del modelo. Para este efecto hacemos uso de la herramienta HHed la cual toma el modelo HMM en cuestión y lo modifica según el número de clases de regresión que queramos. Realizaremos nuestros experimentos empleando 1, 2, 4, 8,16 y 32 clases de regresión.

Para la realización de las pruebas no hemos entrenado un HMM, sino que hemos partido de uno ya entrenado en un proyecto previo del laboratorio ATVS [13]. El modelo fue entrenado con la base de datos TIMIT, tiene un total de 40 gaussianas por estado y consiste en un modelo monofonético.

Otro elemento importante es el modelo de lenguaje. En las pruebas que se habían realizado con YOHO y en las pruebas que realizamos también en este TFG con YOHO, cuando decodificamos un fichero de audio con HTK lo hacemos empleando como modelo de lenguaje una gramática forzada, esto es, no dejamos que se decodifique el audio tal cual, sino que se ha de tener en cuenta una gramática que sólo permite reconocer la secuencia que solicita el sistema. Para ello, partiendo de un fichero de texto se emplea la herramienta HParse para crear la gramática de palabras que básicamente indica el inicio, fin de frase y los caminos posibles entre las palabras del texto.

4.2.2 Protocolo de pruebas

Para la realización de los experimentos con la base de datos YOHO tenemos en cuenta el protocolo seguido por [13]:

- Se utilizan 6 archivos de entrenamiento pertenecientes a una única sesión
- Para la comparación locutor/locutor se emplean todos los ficheros de test del locutor (todos),
- Para la comparación locutor/impostor se emplea un fichero de test del resto de locutores (137), elegidos aleatoriamente.
- Por lo tanto, por cada modelo de locutor tenemos, 40+137=177 pruebas.
- En total tendremos 177 pruebas por los 138 modelos de locutor posibles, 138*177=24426, de los cuales:
    - Enfrentamientos modelo/locutor: 138*40 = 5520.
    - Enfrentamientos modelo/impostor: 138*137 = 18906.

Por ello describiremos a continuación cómo hemos generado los modelos de locutor y cómo hemos realizado la fase de test.

4.2.3 Entrenamiento de los modelos de locutor

Como hemos comentado anteriormente, para generar los modelos de locutor empleamos 6 archivos de entrenamiento pertenecientes a una única sesión. Ésto consta de los siguientes pasos:

- Generación de la gramática del texto: por cada uno de los ficheros disponemos de su transcripción en palabras. Usando la herramienta HParse generamos la gramática correspondiente.
- Cálculo de los vectores de características (MFCCs) de cada fichero, empleando la herramienta HCopy, con la configuración ya comentada.
- Modificación del modelo con la herramienta HHed dependiendo del número de clases de regresión deseado.
- Adaptación del modelo de locutor: partiendo del modelo general, los vectores de características y la gramática generados, creamos el modelo de locutor de la siguiente manera:
    - Decodificación forzada del audio de entrenamiento con la herramienta HVite (decodificación Viterbi), empleando el modelo independiente del locutor.
    - Adaptación *MLLR* del modelo independiente del locutor para crear el modelo dependiente del locutor, con la herramienta HEAdapt.

Con ésto, disponemos de un modelo de locutor con el que podremos decodificar los audios de test en la siguiente fase.

4.2.4 Fase de test

Para la fase de test se parte del protocolo explicado anteriormente. Las fases que la componen son:

- Generación de la gramática del texto: por cada uno de los ficheros de test, de los que disponemos su transcripción en palabras, usamos la herramienta HParse y generamos la gramática correspondiente.
- Cálculo de los vectores de características (MFCCs) de cada fichero, empleando la herramienta HCopy, con la configuración ya comentada.
- Realizamos dos decodificaciones forzadas con HVite de los ficheros de test. Una con el modelo adaptado al locutor y otra con el modelo genérico independiente del locutor, con los ficheros de test correspondientes a cada modelo de locutor. Como

resultado, disponemos de los ficheros (MLF) a partir de los cuales se calcula la puntuación.

4.2.5 Cálculo de scores y decisión

El score final se calcula restando la puntación obtenida con el modelo adaptado a un locutor y la obtenida con el modelo general. A continuación mostramos dos ficheros MLF, resultado de la decodificación para ambos casos (correspondiente a un intento de autenticación de un impostor):

```
#! MLF! #    # speaker dependent
"*/21_64_32.rec"
0 300000 silini -39022.179688
300000 2100000 twenty -23722.453125
2100000 3000000 one -19519.894531
…
```

```
#! MLF! #    # speaker independent
"*/21_64_32.rec"
 0 600000 silini -26137.900391
600000 5000000 twenty -68374.101562
5000000 5900000 one -16801.142578
…
```

El score se calcula, sin tener en cuenta los silencios reconocidos, como la suma global de la diferencia, palabra a palabra, entre las puntuaciones obtenidas del modelo dependiente de locutor y del independiente de locutor. Posteriormente se divide entre la suma global del tiempo total de todas las palabras para obtener una media.

Tendremos que clasificar estas puntuaciones como correspondientes a dos casos distintos:

- Target: el locutor de test coincide con el modelo de locutor.
- Non-target: el locutor de test no coincide con el modelo de locutor.

Para establecer el umbral de decisión obtenemos la curva DET a partir de las puntuaciones calculadas. Éste se puede tomar como el umbral para el punto de igual error (EER), sin embargo, dependiendo de la aplicación final y de lo restrictivos que queramos ser en cuanto al FR y la FA puede ser conveniente elegir otro umbral.

4.3 RSR2015

Para la realización de las pruebas con la base de datos RSR2015 se hará uso de dos herramientas, HTK y Kaldi.

4.3.1 Protocolo de pruebas

Para la realización de los experimentos tomamos la parte 3 de la base de datos, en concreto, la parte que opera con cadenas aleatorias de 10 dígitos. Por ello, el protocolo de pruebas que vamos a seguir es el propuesto en [3]:

- En total tenemos 9 sesiones por locutor, las sesiones correspondientes al entrenamiento son la 1,4 y 7, mientras que las sesiones correspondientes a test son la 2, 3, 5, 6,8 y 9.
- Para la fase de entrenamiento se obtiene un modelo de locutor por sesión, esto es, para las sesiones 1,4 y 7 se entrenan 3 modelos dependientes del locutor respectivamente, con 3 secuencias de 10 dígitos definidas en un fichero proporcionado en la base de datos.
- Para la fase de test se enfrenta a cada modelo de locutor sus correspondientes ficheros de test, definidos en un fichero proporcionado en la base de datos, el cual explicaremos más adelante.

- La base de datos divide a los usuarios en “development” y “evaluation”. Para el caso de “development” tenemos 97 usuarios, esto hace un total de 97*3=291 modelos de locutor, Para el caso “evaluation” disponemos de 106 usuarios, lo que hace un total de 106*3=318 modelos de locutor. En total entrenamos 291+318 = 609 modelos de locutor.
- La base de datos hace distinción entre 3 escenarios posibles:
    - En el primero de ellos se hace una comparación locutor/locutor, esto es, los ficheros de test con los que enfrentamos el modelo de locutor se corresponden al mismo locutor del modelo en ambos casos, para target y non-target. En este caso, el usuario target se corresponde con un locutor cuya locución coincide con el texto solicitado, mientras que el usuario non-target se corresponde con un locutor cuya locución no coincide con el texto solicitado (es decir, es el usuario que dice ser, pero no dice el texto que debería decir). Este caso lo llamaremos *“LOC-LOC”.*
    - En el segundo de ellos se hace una comparación locutor/impostor, esto es, el fichero de test con el que enfrentamos el modelo de locutor corresponde a un locutor distinto del modelo para el caso non-target, mientras que para un usuario target el modelo de locutor coincide con el propio locutor. En este caso, el usuario target se corresponde con un locutor cuya locución coincide con el texto solicitado y el usuario non-target se corresponde con un impostor cuya locución coincide con el texto solicitado también. Este caso lo llamaremos *“LOC-IMPOK”.*
    - En el tercero y último de ellos se hace una comparación locutor/impostor. Sin embargo, la diferencia con el caso anterior reside en que la locución del impostor no coincide con el texto solicitado. Este caso lo llamaremos *“LOC-IMPNOOK”.*

Teniendo en cuenta estos puntos, describiremos cómo se ha implementado la fase de entrenamiento y la fase de test para esta base de datos.

4.3.2 Ficheros de entrenamiento

Para el entrenamiento de los modelos de locutor se parte de los ficheros de entrenamiento proporcionados en la base de datos. Estos ficheros tienen como nombre: *“Xseq10_Y_Z.trn”,* donde X es 3 (3 secuencias de dígitos), Y es “eval” o “dev” según el caso de estudio y Z es “f” o “m” dependiendo si el locutor es hombre (male) o mujer (female). Los ficheros incluyen lasiguiente información (se muestra un ejemplo):

“m067_04 male/m067/m067_04_061.sph male/m067/m067_04_062.sph male/m067/m067_04_063.sph”

En este caso, para entrenar el modelo de locutor correspondiente al usuario m067_04, donde 04 es la sesión (puede tomar valor 01,04 o 07), se tienen en cuenta 3 audios. Estos 3 audios vienen indicados por el número de sesión (04) y por el identificador de la frase (061,062 y 063 respectivamente). Las transcripciones de las palabras correspondientes a esos audios las obtenemos de un fichero llamado *“promptpart3.lst”,* donde vienen indexadas por número de sesión e identificador de la frase.

De esta forma, podemos obtener la transcripción de la secuencia de dígitos teniendo en cuenta los números que aparecen en ella (la obtenemos mediante un programa propio escrito en Python):

"2-8-0-3-6-5-4-1-9-7 → TWO-EIGHT-ZERO-THREE-SIX-FIVE-FOUR-ONE-NINE-SEVEN"

Por ello, teniendo en cuenta los ficheros de entrenamiento proporcionados, obtenemos un modelo de locutor para cada caso en concreto, empleando los audios correspondientes para adaptar al locutor el modelo genérico independiente del locutor.

4.3.3 Ficheros de test

Para la fase de test, y teniendo en cuenta los tres escenarios descritos anteriormente (LOC-LOC, LOC-IMPOK, LOC-IMPNOOK), se toman los ficheros de test proporcionados en la base de datos, del tipo: *"seqX_testY_Z.ndx"*, donde X toma el valor 10 (secuencias de 10 dígitos), Y es "eval" o "dev" y Z es "f" o "m", dependiendo si el locutor es hombre (male) o mujer (female). De este modo, los ficheros de test incluyen la siguiente información:

"1-7-4-0-9-3-8-2-5-6, f067_04, f067/f067_08_061.sph"
"text-prompted","modelo de locutor","audio de test"

que consta de tres elementos:

- Text-prompted: texto que solicita el sistema. Para acceder a éste es necesario que la locución dicha por un usuario coincida con ésta.
- Modelo de locutor: del total de modelos entrenados en la fase de entrenamiento, se enfrenta el audio de test con este modelo.
- Audio de test: Audio grabado del usuario. Si el usuario coincide con el modelo de locutor estamos en una comparación *LOC-LOC*, en este caso sí coinciden ya que el modelo es f067_04 y el audio de test pertenece al usuario f067, El resto de elementos que acompañan al audio de test se corresponden con la frase dicha, esto es, 08_061 se corresponde con una secuencia de dígitos determinada. Si ésta coincide con el text-prompted y además es el locutor estamos ante un caso target del escenario *LOC-LOC*. Sin embargo, en este ejemplo lo dicho en el audio de test es "7-2-9-8-0-1-6-4-3-5", que no coincide con el text-prompted, por lo tanto estamos ante un caso non-target del escenario *LOC-LOC*.

A modo de resumen, y teniendo en cuenta los tres escenarios descritos tenemos:

- Escenario LOC-LOC:

  "6-0-8-4-7-5-9-1-3-2, f067_04, f067/f067_02_061.sph" → TARGET
  "1-7-4-0-9-3-8-2-5-6, f067_04, f067/f067_08_061.sph" → NON-TARGET

  En el caso target, el locutor coincide con el locutor del modelo y el texto solicitado y el audio coniciden, mientras que en el caso non-target el locutor coincide con el modelo pero el texto solicitado y el audio de test no coinciden.

- Escenario LOC-IMPOK:

  "6-0-8-4-7-5-9-1-3-2, f067_04, f067/f067_02_061.sph" → TARGET
  "6-0-8-4-7-5-9-1-3-2, f067_04, f048/f048_02_061.sph" → NON-TARGET

El caso target es idéntico al explicado anteriormente. En el caso non-target, el locutor de test no coincide con el modelo pero sin embargo la frase dicha en el audio coincide con el texto solicitado.

- Escenario LOC-IMPNOOK:

  "6-0-8-4-7-5-9-1-3-2, f067_04, f067/f067_02_061.sph" → TARGET
  "6-0-8-4-7-5-9-1-3-2, f067_04, f087/f087_08_061.sph" → NON-TARGET

  El caso target es idéntico al explicado anteriormente. En el caso non-target, el locutor de test no coincide con el modelo y además la frase dicha por éste tampoco coincide con el texto solicitado.

4.3.4 Descripción general y modelos empleados en HTK

Para realizar los experimentos con la base de datos RSR2015 partimos de lo ya implementado para la base de datos YOHO, teniendo en cuenta lo siguiente:

- Se utilizarán 32 clases de regresión, ya que por el proyecto [13], se conseguían los mejores resultados.
- El modelo empleado es exactamente igual al utilizado con YOHO.
- No se ha modificado el modelo de lenguaje utilizado.
- La adaptación de los modelos del locutor, decodificación y cálculo de scores se realizan empleando los mismos mecanismos que con YOHO.
- La configuración para extraer los MFCCs es idéntica.

4.3.5 Entrenamiento en HTK

Para el entrenamiento de los modelos de locutor de la base de datos RSR2015 con HTK se ha tenido en cuenta el protocolo de entrenamiento explicado anteriormente. Los pasos para implementar este módulo son idénticos a aquellos realizados con la base de datos YOHO, los cuales hemos analizado previamente.

4.3.6 Test en HTK

Para la fase de test se tiene en cuenta el protocolo de test previamente analizado, diferenciando tres casos. La creación de la gramática y de los vectores de características de los ficheros de test sigue el mismo esquema que empleábamos para la base de datos YOHO.

4.3.7 Cálculo de scores y decisión

Los scores se calculan de la misma manera que hacíamos con los datos de YOHO. A continuación vemos un ejemplo de dos fragmentos de ficheros MLF resultado de la decodificación (para el caso de un locutor que dice bien la frase):

```
#! MLF! # #speaker dependent
"*/m147_02_061.rec"
0 300000 silini -18103.291016
300000 2200000 six -69114.062500
2200000 40200000 zero -1108386.000000
…
```

```
#! MLF! # # speaker independent
"*/m147_02_061.rec"
0 300000 silini -79496.960938
300000 2200000 six -56202.558594
2200000 40200000 zero -18994.085938
…
```

Con ésto, calculamos las puntuaciones target y non-target, teniendo en cuenta los 3 escenarios posibles de los que disponemos. A partir de estas puntuaciones obtendríamos la curva DET con la que evaluamos los resultados del sistema. Para establecer un umbral de decisión concreto, tendríamos que analizar los requisitos de la aplicación concreta.

4.3.8 Descripción general y modelos empleados en Kaldi

Debido a que el sistema de reconocimiento de voz que empleábamos con HTK era muy básico, se decidió emplear una herramienta de reconocimiento de voz distinta a éste, que permitía más flexibilidad de programación de scripts y nos proporcionaba la posibilidad de emplear modelos fonéticos más sofisticados y entrenados con más horas de voz que los que usábamos con HTK.

El modelo empleado para la realización de los experimentos ha sido tomado del trabajo realizado por el grupo ATVS. Este modelo ha sido entrenado con datos de la base de datos Switchboard. El modelo ha sido generado a partir del script *"train_sat.sh",* proporcionado por la herramienta KALDI, y que usa 4000 estados y 100000 gaussianas. Además, se trata de un modelo trifonético con adaptación al locutor por frase.

Para la creación de los MFFCs se emplea la siguiente configuración:

- Ventana tipo hamming.
- Filtro de frecuencia de corte inferior de 64 Hz y superior de 3800 Hz.
- Número de coeficientes cepstrales por trama: 13.

Los MFCCs se generan a partir de un script ya proporcionado por la herramienta (*"make_mfcc.sh"*).

Además, tras la creación de los vectores de características se aplica a los parámetros CMVN (cepstral mean variance normalization), que consiste en la normalización de la distribución de los parámetros de voz de entrada, forzando éstos a tener media 0 y varianza 1, y consiguiendo aliviar el problema de variabilidad de sesión.

Para la preparación de los datos Kaldi necesita que se generen unos ficheros determinados:

- "Text": contiene el texto, indexado por <utterance-id> <text>, esto es, identificación de la locución y transcripción.
- "wav.scp": viene indexado por <utterance-id> <ruta-del-fichero-wav>, esto es, relaciona la identificación de la locución con la ruta en disco de su fichero wav correspondiente.
- "utt2skp": indexado por <utterance-id> <speaker>, relaciona la identificación de la locución con el locutor que la ha generado.
- "spk2utt": similar al anterior, indexado por locutor e identificación de la locución o locuciones que se corresponden con él.

4.3.9 Generación del modelo de lenguaje

Debido a que la base de datos SWBD (Switchboard) ha sido entrenada con frases en inglés pertenecientes a un entorno de voz telefónica conversacional (donde raramente se van a dar secuencias de 10 dígitos), se creado un nuevo modelo de lenguaje, con el fin de adecuarlo a

la tarea de decodificación de secuencias de dígitos y además de reducir al máximo la tasa de error de palabra o WER.

Para ello, hemos generado 270.000 secuencias aleatorias de 10 dígitos cada una (en cada secuencia tenemos todos los números del 0 al 9), con un programa propio en Python. El resultado es el siguiente:

"s1-1-2-5-9-8-6-0-3-7-4 ONE TWO FIVE NINE EIGHT SIX ZERO THREE SEVEN FOUR
…
s269999-3-9-6-8-2-0-5-7-1-4 THREE NINE SIX EIGHT TWO ZERO FIVE SEVEN ONE FOUR".

Teniendo este texto sintético, nos disponemos a crear el modelo de lenguaje y su grafo correspondiente con los scripts ya proporcionados por la herramienta Kaldi (*"train_lms.sh"* y *"mkgraph.sh"*). Con esto, tenemos listo nuestro modelo de lenguaje para proceder a la fase de entrenamiento y test.

Para evaluar el rendimiento en cuanto WER con el nuevo modelo de lenguaje, se toman todas las locuciones de entrenamiento de todos los locutores pertenecientes a los dos grupos ya mencionados de la base de datos RSR2015.

4.3.10 Entrenamiento en Kaldi

Para la fase de entrenamiento se sigue el protocolo de pruebas comentado anteriormente para esta base de datos. El esquema que sigue el entrenamiento de los modelos de locutor se puede ver en la siguiente figura:

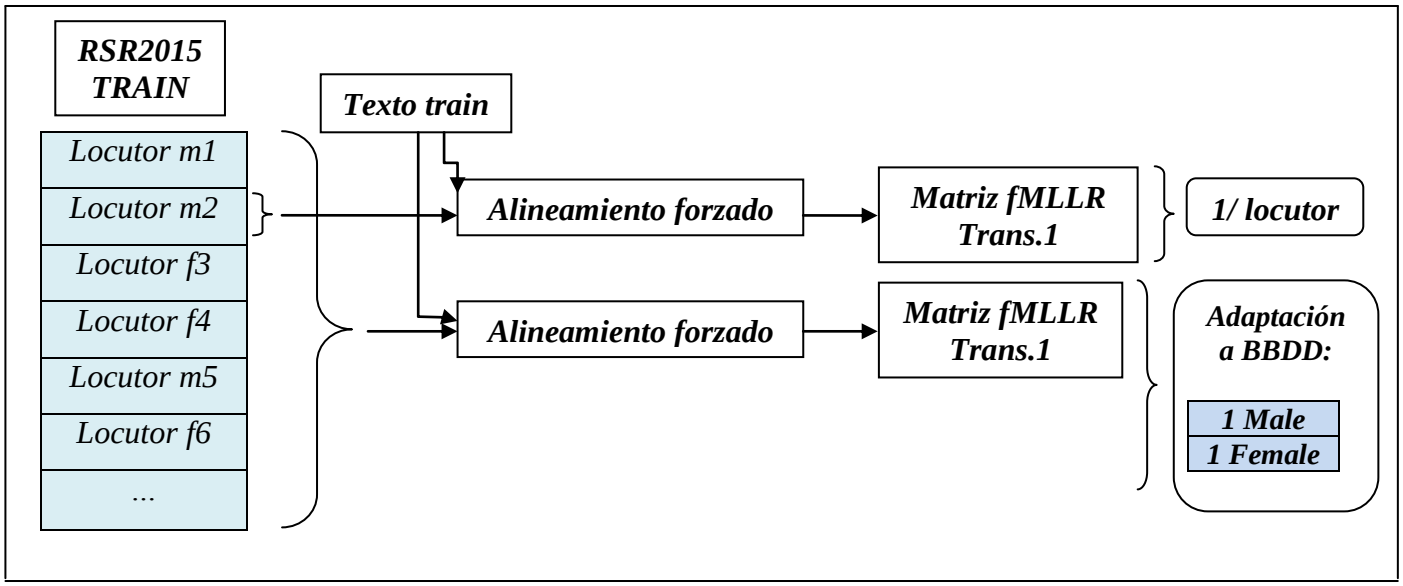

Figura 4-1: Entrenamiento de los modelos de locutor

Las matrices de transformación entrenadas son:

- Tomando todos los datos de entrenamiento de todos los locutores masculinos que pertenecen a la parte "development" y "evaluation", se entrena una matriz general de adaptación de la base de datos RSR2015 para locutores masculinos.

- Tomando todos los datos de entrenamiento de todos los locutores femeninos que pertenecen a la parte “development” y “evaluation”, se entrena una matriz general de adaptación de la base de datos RSR2015 para locutores femeninos.
- Tomando los datos de entrenamiento pertenecientes a un modelo de locutor dado, se entrena una matriz de adaptación para cada modelo dado. Esta matriz constituye el modelo dependiente del locutor.

Los pasos que se han tomado para generarlas han sido:

- Determinar los datos de train para cada uno de los tres casos mencionados anteriormente.
- Con esos datos y sus correspondientes textos, se obtiene la matriz de transformación a partir de un alineamiento forzado que emplea fMLLR. Ésto se realiza a partir de un script proporcionado por la herramienta Kaldi (*“align_fmllr.sh”*).

4.3.11 Test en Kaldi

Para la fase de test se distinguen dos escenarios. El primero se centra en la verificación de locutor, mientras que el segundo se centra en la verificación de la frase. Para que un usuario determinado acceda al sistema ambos bloques (verificación de locutor y frase), deben aceptarlo.

4.3.11.1 Verificación del locutor

La verificación de locutor sigue el siguiente esquema:

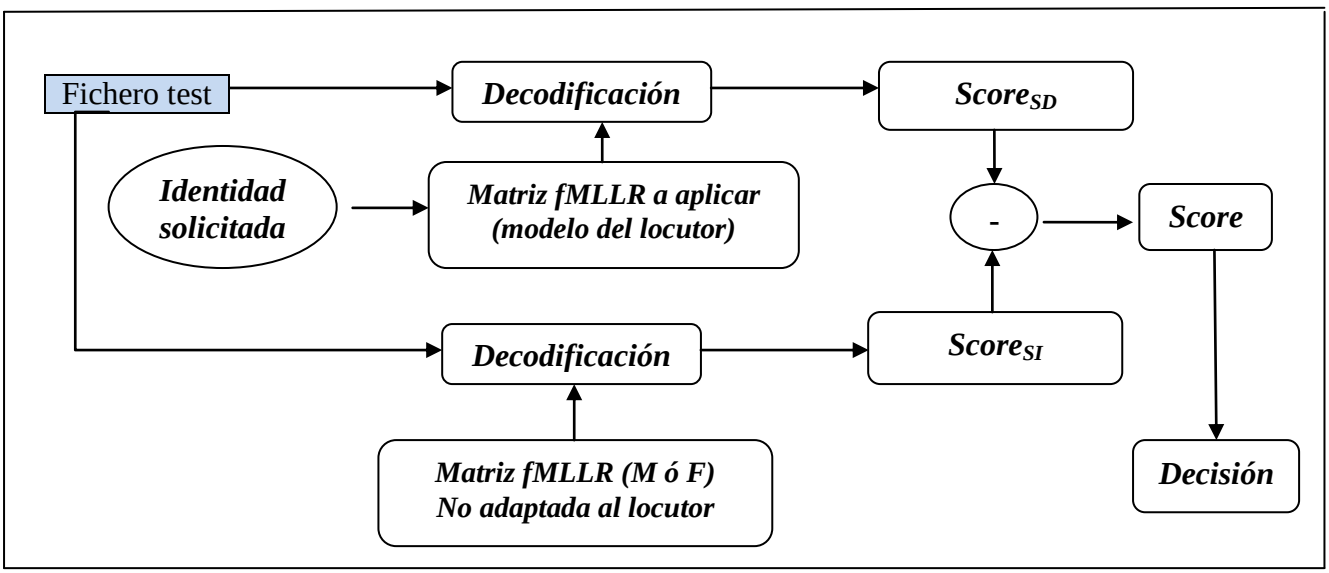

Figura 4-2: Verificación del locutor

Las entradas al sistema son, un fichero de test y una identidad solicitada. De este modo se realizan dos decodificaciones en paralelo. La primera toma la matriz fMLLR del modelo de locutor que se corresponde con la identidad solicitada, mientras que el segundo toma una matriz fMLLR en función del sexo de la identidad reclamada. Con ello, se calculan dos puntuaciones (solo se tiene en cuenta la puntuación acústica, no la del modelo de lenguaje), una dependiente del modelo del locutor y otra independiente del locutor. Ésto se

realiza mediante una decodificación Viterbi de máxima verosimilitud (se toma el mejor camino), sin tener en cuenta el texto (no se hace una decodificación forzada, esto es, otorgamos libertad al sistema en el reconocimiento). Ambas puntuaciones se restan (son puntuaciones en escalas logarítmicas) y se obtiene la puntuación final. A continuación se toma una decisión (rechazar o aceptar a ese locutor que reclama una identidad en concreto) en función de un umbral determinado.

Las puntuaciones parciales se toman de los ficheros de salida del proceso de decodificación, llamados *"decode.log",* donde encontramos para cada locución el logaritmo de la verosimilitud por frame. A continuación mostramos un ejemplo de una locución en concreto:

"f050_06_061 SEVEN ZERO THREE FOUR EIGHT NINE ONE SIX TWO FIVE
Log-like per frame for utterance f050_06_061 is -4.38435 over 577 frames."

En la primera línea vemos la secuencia de dígitos decodificada, mientras que en la segunda tenemos la puntuación obtenida en una serie de frames determinados.

4.3.11.2 Verificación de la frase

La verificación de la frase tiene como objetivo aceptar o rechazar a un usuario en función de si el texto solicitado por el sistema coincide con lo que el usuario dice, si no coinciden se rechazará al usuario, mientras que si coinciden se le aceptará (siempre y cuando el bloque de verificación de locutor acepte también al usuario). La verificación de la frase sigue el siguiente esquema:

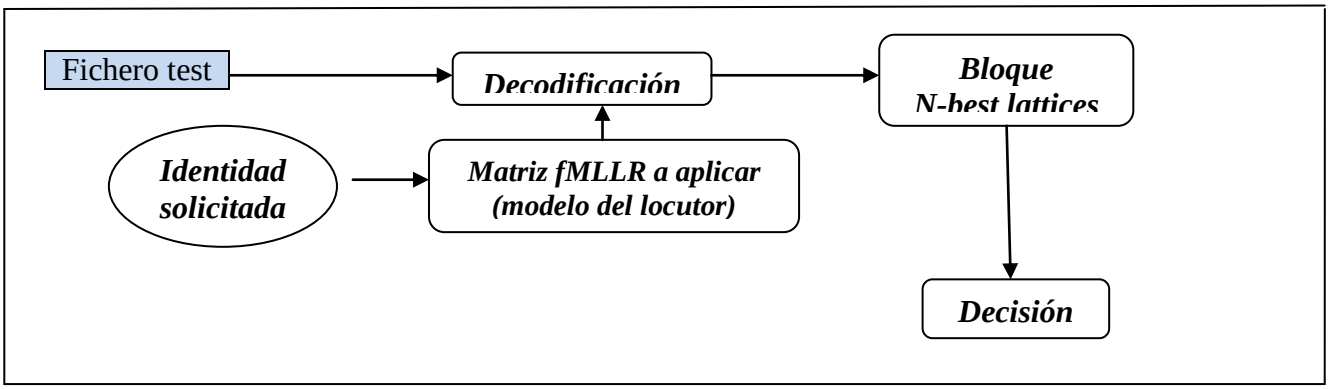

Figura 4-3: Verificación de la frase

Las entradas al sistema son de nuevo un fichero de test y una identidad solicitada. Para la identidad solicitada se toma su matriz fMLLR correspondiente y con ella se realiza una decodificación no forzada generando un lattice que contiene múltiples hipótesis de reconocimiento. A continuación tenemos el bloque *N-best lattices* cuyo objetivo es, por cada locución, tomar los *N* mejores caminos que aparezcan en el lattice resultantes de la decodificación. Con ello, el texto solicitado se busca en esos *N* mejores caminos, si no se encuentra el locutor será rechazado, mientras que si se encuentra, este bloque aceptará al locutor. Por último, podemos ser menos exigentes en cuanto a la coincidencia exacta del texto solicitado y el texto reconocido, es decir, se daría la posibilidad de aceptar a un usuario aunque el reconocedor haya cometido un cierto número de errores como máximo, lo que permite reducir la tasa de falsos rechazos por este motivo, tratando de mantener controlada la tasa de faltas aceptaciones.

5 Integración, pruebas y resultados

5.1 Resultados YOHO con HTK

A continuación se muestra la curva DET (CR se corresponde con clases de regresión empleadas) obtenida con nuestras pruebas, y la curva DET obtenida en el trabajo de partida del grupo ATVS en el que nos basábamos [13], resultado ambas de los experimentos con la base de datos YOHO:

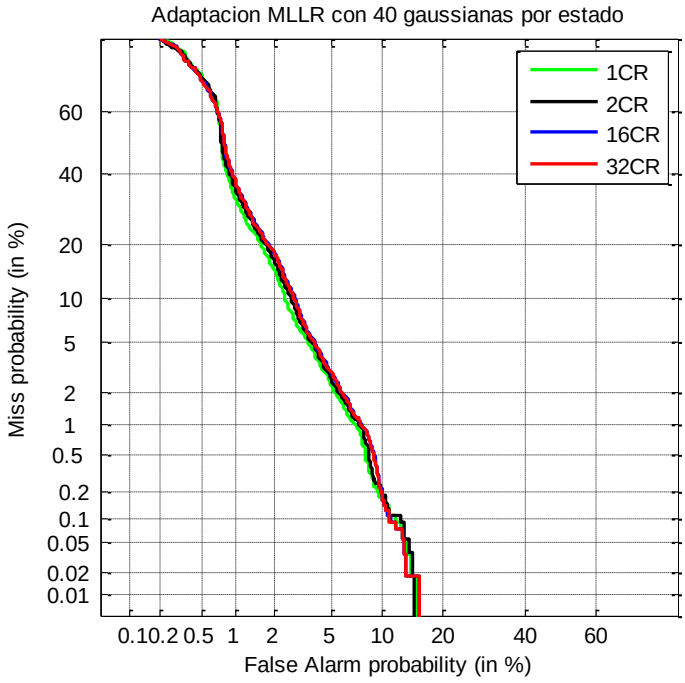

Figura 5-1: Resultados de verificación de locutor con YOHO con HTK obtenidos en este trabajo

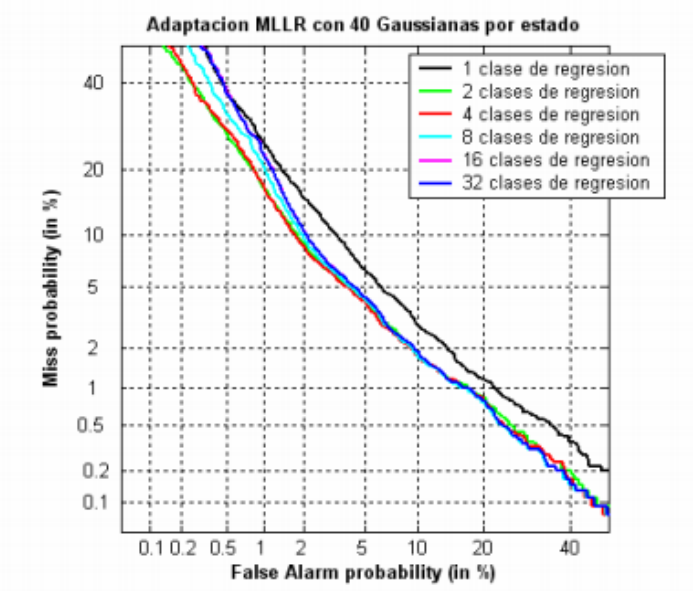

Figura 5-2: Resultados de verificación de locutor con YOHO y HTK obtenidos en el trabajo de partida para este TFG [13]

Comparando las dos curvas DET anteriores podemos ver que se obtienen resultados similares (en ambos experimentos el EER está ligeramente por debajo del 5%).

5.2 Resultados RSR2015

A continuación presentamos los resultados obtenidos en los experimentos con la base de datos RSR2015. Haremos distinción entre aquellos obtenidos con HTK y aquellos obtenidos con Kaldi.

5.2.1 HTK

En el experimento realizado con HTK se tienen en cuenta los dos conjuntos de la base de datos ("development" y "evaluation"), para todos los locutores de éstos, sin discriminar entre masculino o femenino, siguiendo el protocolo de pruebas comentado en la sección *4.3.11.1* de este trabajo.
La curva DET obtenida con HTK para los casos de análisis *LOC-IMPOK* (locutor e impostor que dice bien el texto mostrado) y *LOC-IMPNOOK* (locutor e impostor que dice mal el texto mostrado) es la siguiente:

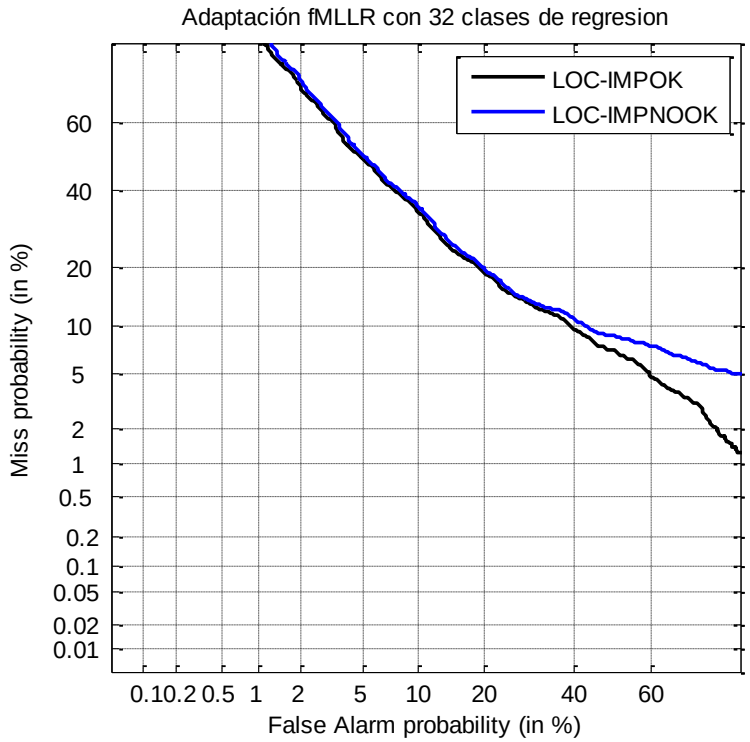

Figura 5-3: Resultados de verificación de locutor con HTK y RSR2015

El EER está ligeramente por debajo del 20%. Comparando con los resultados de la siguiente tabla para un sistema i-vector, se puede ver que se consiguen resultados muy próximos, los cuales están entre el 16.37% y el 18.56% para locutores masculinos y femeninos respectivamente (en nuestros resultados están considerados ambos géneros conjuntamente):

User	Target		Impostor		Male		Female	
Text	Correct	Wrong	Correct	Wrong	HiLAM	*i*-vector	HiLAM	*i*-vector
Trials	tar	non	–	–	38.32/99.96		38.35/98.18	
	tar	–	non	–	6.50/33.39	16.37/69.09	10.60/44.30	18.56/80.78
	tar	–	–	non	6.13/29.84		10.55/40.00	

Figura 5-4: Resultados de verificación de locutor para los tres casos de análisis: enfrentamiento LOC-LOC, correspondiente a la primera línea, enfrentamiento LOC-IMPOK, correspondiente a la segunda línea y enfrentamiento LOC-IMPOOK correspondiente a la tercera línea. Se hace una división entre hombres y mujeres, y entre un sistema HiLAM e i-vector. Los números que aparecen siguen este formato (EER %/minDCF 100) [3]

5.2.2 Kaldi

Para los experimentos realizados con Kaldi, nos centramos en dos puntos. El primero de ellos es la verificación de locutor mientras que el segundo trata de la verificación de la frase.

5.2.2.1 Verificación de locutor

Para los experimentos de verificación de locutor se toman todos los locutores pertenecientes a los dos conjuntos de la base de datos. El protocolo de pruebas es idéntico al empleado en HTK, con la salvedad que se utilizan matrices de transformación adaptadas a la base de datos para el cálculo de puntuaciones independientes del locutor, como ya se comentó en la sección *4.3.10*.

La curva DET obtenida con Kaldi para los casos LOC-IMPOK y LOC-IMPNOOK es la siguiente:

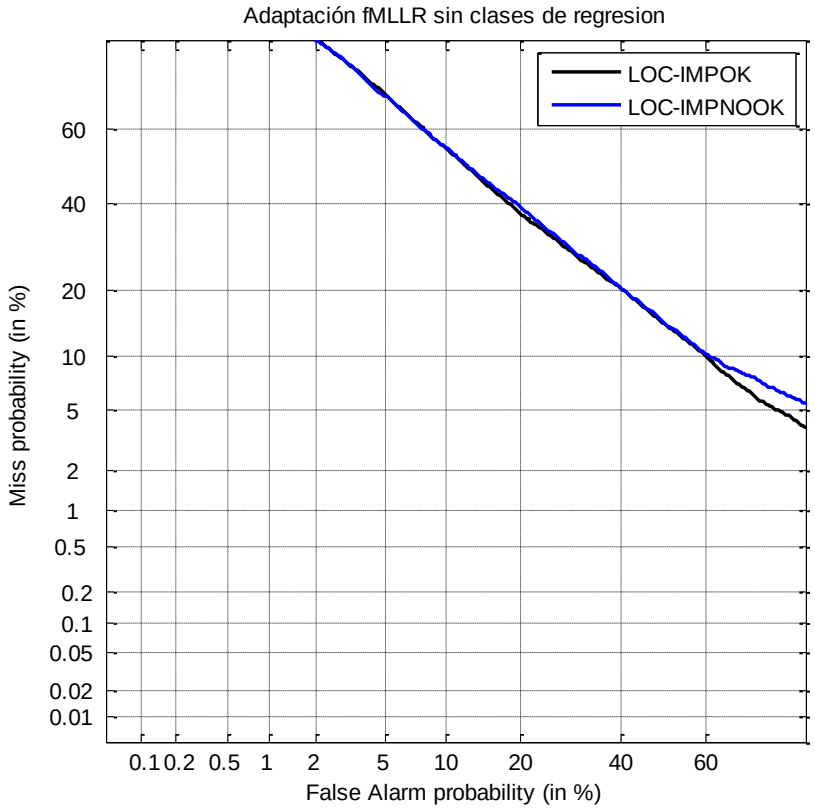

Figura 5-5: Resultados de verificación de locutor con Kaldi y RSR2015

Donde el EER está entorno al 30%, por lo tanto el resultado empeora respecto a lo obtenido con HTK. Para analizar lo que está ocurriendo, vamos a obtener distintas curvas DET para algunos modelos de locutor de ejemplo.
En total hemos escogido 18 locutores, divididos entre masculinos y femeninos. A continuación se muestran algunas curvas DET para ciertos modelos de locutor:

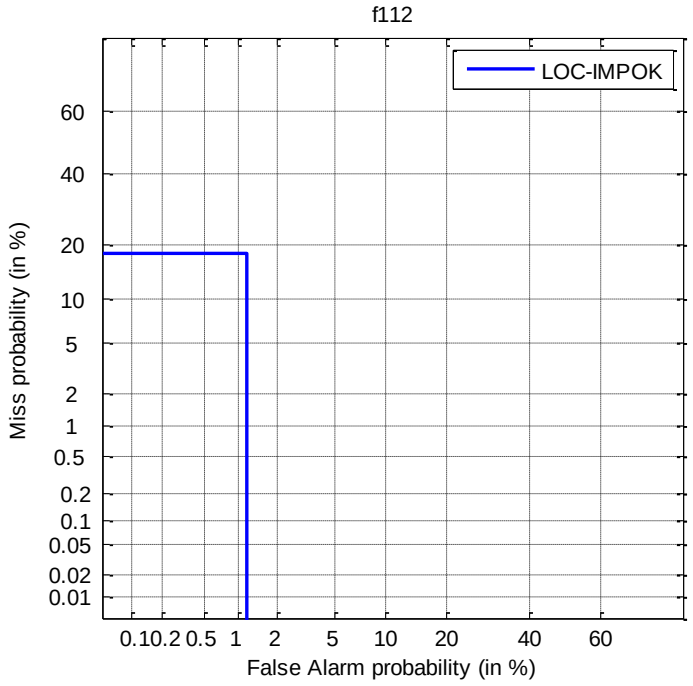

Figura 5-6: Resultados de verificación de locutor para el usuario f112

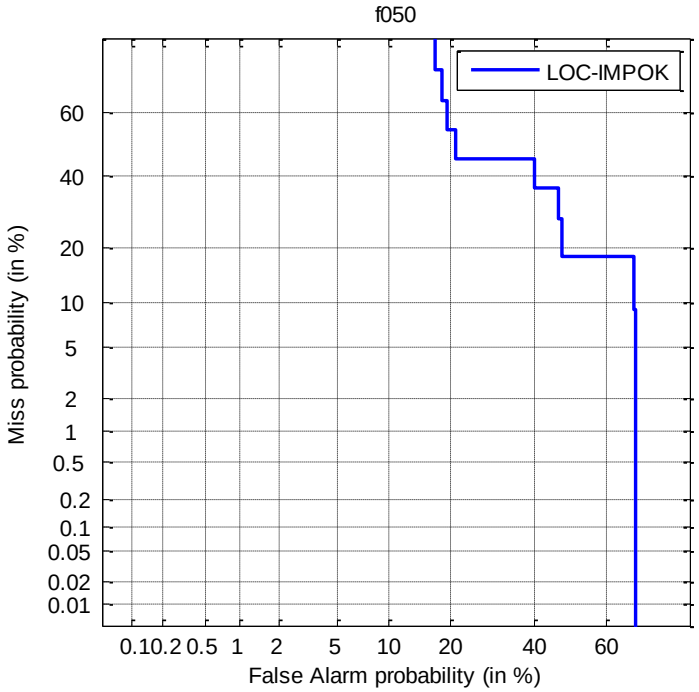

Figura 5-7: Resultados de verificación de locutor para el usuario f050

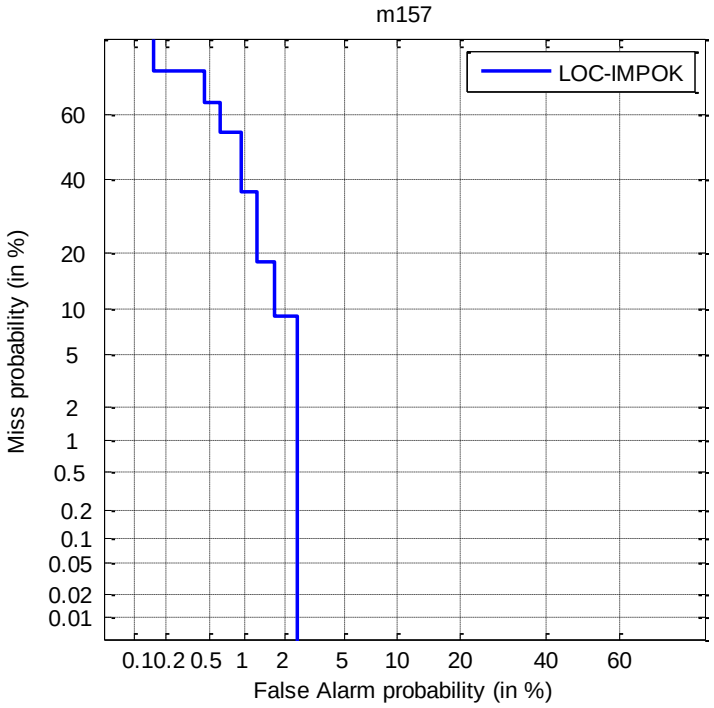

Figura 5-8: Resultados de verificación de locutor para el usuario m157

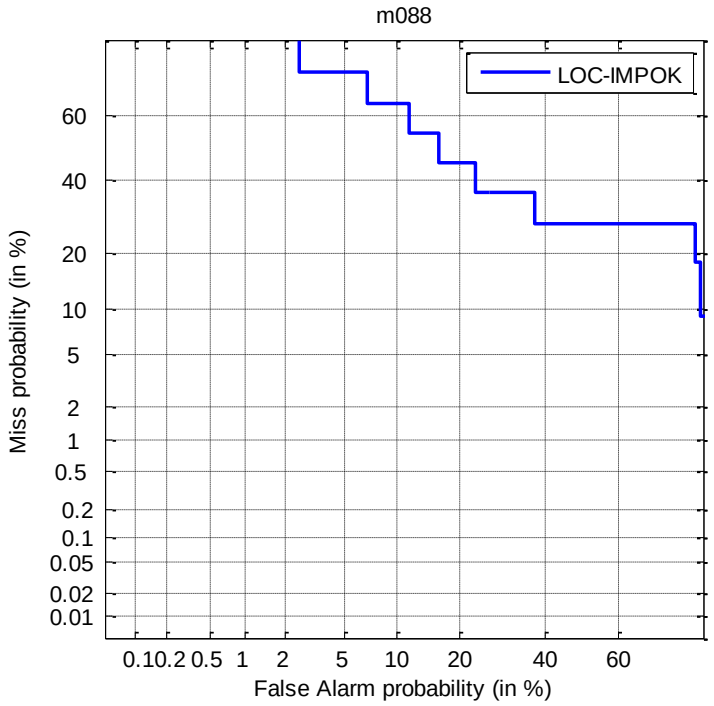

Figura 5-9: Resultados de verificación de locutor para el usuario m088

Como se puede observar, hay modelos de locutor donde el sistema funciona bastante bien (para el primer modelo femenino tenemos un EER de 1.5% aproximadamente y para el primero masculino de 2%), mientras que en otros no (para el caso femenino y masculino el EER es aproximadamente 40% en el segundo caso mostrado). La curva DET global para los 18 modelos analizados en detalle es similar a la obtenida para toda la base de datos:

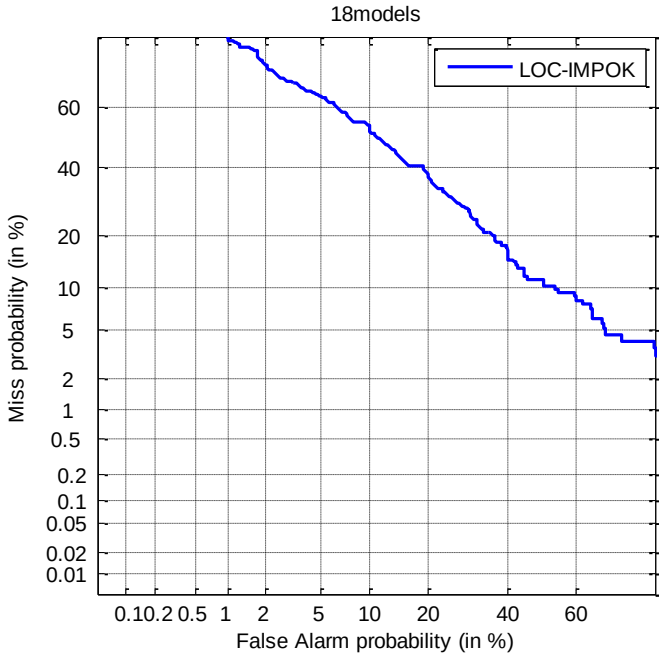

Figura 5-10: Resultados de verificación de locutor con 18 usuarios

Por lo tanto, nos encontramos con modelos donde hay buenos resultados en cuanto a EER se refiere, mientras que en otros los resultados son muy malos. Una hipótesis para este mal resultado en algunos casos es que las puntuaciones que se obtienen para cada locutor no están siempre bien alineadas (las curvas DET de un locutor que se han mostrado anteriormente se obtienen en realidad con tres modelos), pero a fecha de cierre de esta memoria no se ha podido todavía comprobar esta hipótesis. En cualquier caso, los buenos resultados que se consiguen con algunos de estos 18 locutores elegidos al azar nos animan a seguir trabajando en esta línea.

5.2.2.2 Verificación de la frase

Tal y como se explicó en la sección *4.3.9* de este trabajo, se crea un nuevo modelo de lenguaje para la tarea de reconocimiento de 10 dígitos, con el fin de reducir el WER al máximo. Por ello realizamos dos pruebas para evaluarlo, una con el modelo de lenguaje modificado y el otro sin modificar (el creado a partir de la base de datos Switchboard). Los resultados obtenidos en porcentaje de error de palabra para la tarea de reconocimiento de 10 dígitos son los siguientes:

	Modelo de lenguaje modificado	***Modelo de lenguaje no modificado***
·%WER	5.30 [1851/34920]	35.01 [12226/34920]
Insercciones	855	1753
Borrados	282	4414
Sustituciones	714	6059

Tabla 5-1: Resultados de reconocimiento de palabra con dos modelos de lenguaje

Como se puede observar, la modificación del modelo de lenguaje conlleva una reducción significativa en cuanto a WER se refiere, pasando de más del *35%* al *5.3%*. Un WER del

5.3% se traduce aproximadamente en un error de frase del *50%*, ya que la frase está compuesta por 10 palabras. Este porcentaje de error de frase es demasiado elevado para realizar una verificación de frase empleando directemante el resultado del reconocimiento. Por ello, para la realización de los experimentos de verificación de frase se procede de la siguiente manera:

- Se calculan *N*-hipótesis de reconocimiento posibles en lugar de solo tener en cuenta la mejor, siendo *N* en nuestras pruebas de 1,10,15,25,50,75 y 100.
- Entre esas posibles *N* hipótesis se busca el texto solicitado por el sistema, la cadena de 10 dígitos.
- A fin de flexibilizar más el compromiso entre falsas aceptaciones y falsos rechazos, se introduce una variable adicional a la que llamaremos *matches*, que indica el mínimo número de aciertos de dígitos que se deben dar en una hipótesis analizada para aceptar o rechazar a un locutor.

Para validar esta metodología hemos realizado pruebas de verificacion de la frase en el escenario *LOC-LOC* (es decir el scenario en el que el locutor de la frase de test coincide con el locutor del modelo, pero en el que el texto en la frase de test puede (target) o no (non-target) coincidir con el del texto solicitado).

Para una única hipótesis de reconocimiento posible obtenemos la siguiente gráfica, que compara la FR y FA en función del número de matches:

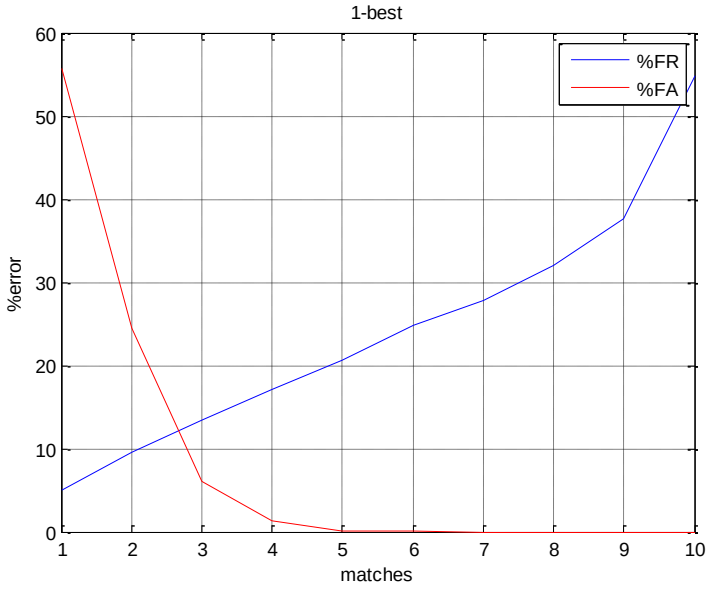

Figura 5-11: Resultados de verificación de frase para 1-best

1-best

#matches	1	2	3	4	5	6	7	8	9	10
FA(%)	59.77	24.51	6.13	1.40	0.05	0	0	0	0	0
FR(%)	4.93	9.52	13.38	17.07	20.53	24.79	27.80	31.98	37.64	54.85

Tabla 5-2: Resultados de verificación de frase para 1-best

El EER está en torno al 12%, sin embargo, la variable matches es de tipo discreto, por lo tanto, el match que consigue una combinación más baja de FA (6.13%) y FR (13.38%) es 3.
Al considerar los resultados 100-best del reconocedor se obtiene la siguiente gráfica:

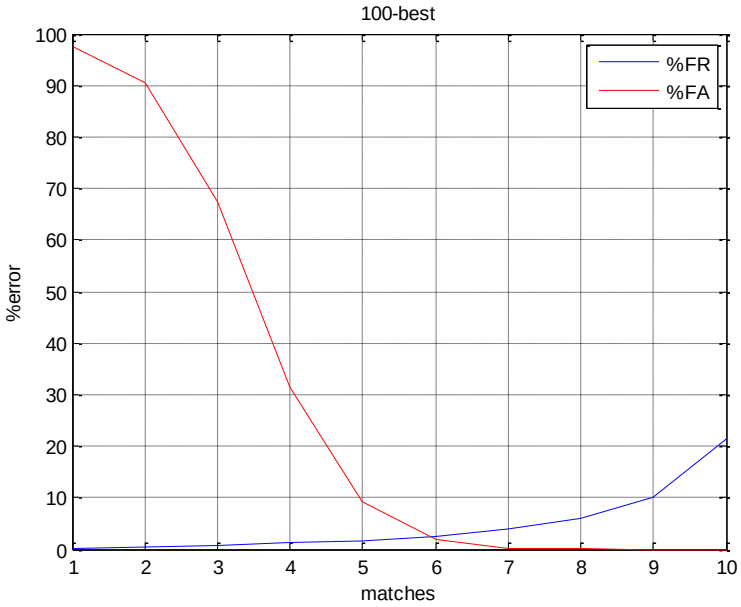

Figura 5-12: Resultados de verificación de frase para 100-best

100-best

#matches	1	2	3	4	5	6	7	8	9	10
FA(%)	97.59	90.50	67.49	31.54	9.32	1.89	0.15	0	0	0
FR(%)	0.19	0.45	0.71	1.17	1.66	2.48	3.87	6.08	10.15	21.44

Tabla 5-3: Resultados de verificación de frase para 100-best

Se aprecia una mejora sustancial sobre los resultados obtenidos con 1-best. Además, el EER es aproximadamente 2% (entre los matches 5 y 6), de esta manera, el match que da mejores resultados en cuanto a FR y FA es 6. En la tabla se puede ver que con 6 matches conseguimos una FA de 1.89% y un FR de 2.48%. Teniendo en cuenta que nuestro reconocedor reconocía el 50% de frases de 10 dígitos de forma incorrecta, este resultado es un muy bueno, e indica que no es necesario tener un reconocedor de voz perfecto para tener un EER más que razonable en la verificación de frases.

Comparando con los resultados equivalentes publicados por los creadores de la base de datos RSR2015 de la figura *5-4* (la primera línea, que en nuestro caso denominamos *LOC-LOC*), vemos que nuestros resultados de EER próximos al 2% son muy superiores a los resultados por encima del 38% de EER publicados por los creadores de la base de datos. Aunque por motivos de tiempo no ha sido posible realizar experimentos de verificación de frase para el caso *LOC-IMPNOOK* (mismo caso que el analizado salvo porque los casos non-target no coinciden ni en identidad del locutor ni en el texto solicitado), es previsible que los resultados de verificación de frase que obtuviésemos fuesen similares a los obtenidos para el caso *LOC-LOC*, por lo que sería probable, únicamente verificando la frase solicitada, mejorar los resultados publicados por los creadores de la base de datos (Figura 5-4) para este caso (tercera línea de la figura), que están entre el 6 y el 10% de EER.

6 Conclusiones y trabajo futuro

Con la realización de este trabajo hemos evaluado un sistema de reconocimiento de locutor preexistente en el grupo sobre una base de datos antigua (YOHO) y sobre una base de datos moderna (RSR2015). Además, hemos implementado un nuevo sistema de reconocimiento de locutor dependiente de texto basado en una nueva herramienta de reconocimiento de voz (Kaldi), con el que hemos hecho experimentos de verificación del locutor y de verificación de frase sobre la base de datos RSR2015.

En primer lugar, con la realización de nuestros experimentos con la base de datos YOHO y el sistema preexistente basado en HTK se han conseguido resultados similares a los obtenidos previamente por el grupo ATVS, que era el objetivo propuesto en este trabajo para asegurar que estábamos empleando correctamente el sistema preexistente.

Siguiendo la línea marcada por los trabajos anteriores del grupo, intentamos replicar el sistema para la nueva base de datos RSR2015, consiguiendo unos resultados muy inferiores a los obtenidos con la base de datos YOHO (el EER de ésta está entorno al 5% mientras que para RSR2015 el EER obtenido está entorno al 20%). Es por ello que nos planteamos la utilización de la herramienta Kaldi, además de unos modelos acústico-fonéticos más avanzados que modelaran mejor la acústica de la voz, con el objetivo de reducir al máximo el EER. Sin embargo, hasta el momento de cierre de esta memoria, no se ha conseguido bajar el 20% de EER obtenido con el sistema preexistente, sino que los resultados globales han resultado peores (30% de EER), utilizando Kaldi. Para analizar el por qué de este peor resultado cuando la herramienta es más avanzada y los modelos acústico-fonéticos mejores hemos realizado un análisis de los resultados por locutores encontrando que para algunos funcionan muy bien (con EERs por debajo del 2%) y otros para los que funciona muy mal (con EERs por encima del 40%). Tenemos varias hipótesis sobre las causas de estos resultados y varias ideas de mejora (como eliminar de la puntuación los silencios como ya se hizo con una notable mejora con el sistema previo), pero al cierre de esta memoria no ha sido posible encontrar una causa única para este resultado. Otra de las hipótesis es que las puntuaciones estén mal alineadas para distintos modelos (en las pruebas que hemos realizado cada locutor incluía 3 modelos, por lo que es posible que para un único locutor haya puntuaciones desalineadas). El análisis detallado de estos resultados y la verificación de estas hipótesis queda como trabajo futuro.

Unos de los resultados más exitosos de este TFG y que no habían sido previamente explorados en el grupo ATVS son los resultados de verificación de frase. Partiendo de un reconocedor de voz con un 5.3% de tasa de error de palabra y aproximadamente un 50% de tasa de error de frase se ha conseguido alcanzar cerca de un 2% de EER, en el mismo caso exactamente en el que los creadores de la base de datos RSR2015 publicaron un resultado de más del 38% de EER. Este resultado, quizás susceptible de mejorarse algo más todavía en el futuro, muestra que este TFG ha dado un gran paso hacia la resolución del problema de la verificación de frase.

Como trabajo futuro se propone la realización de experimentos empleando redes neuronales profundas en los modelos acústico-fonéticos, de las que se sabe que funcionan mejor que los HMMs, con el objetivo de mejorar los resultados obtenidos.
Además, la base de datos RSR2015 es bastante extensa, ya que como explicamos en la sección *3.3.2.2*, consta de partes relacionadas con comandos de voz para hogares inteligentes, en la que se propone el entrenamiento y test de los modelos, así como otra parte donde se dan secuencias de 5 dígitos. Con esto, se abre el camino para seguir trabajando con esta nueva base de datos en otras tareas no abordadas todavía.

Referencias

[1] Benesty, J., Sondhi, M. M., & Huang, Y. (Eds.) (2007). *Springer handbook of speech processing, Chapter 37- Text-dependent speaker recognition.* Springer Science & Business Media.

[2] González-Rodríguez, J., Toledano, D. T., & Ortega-García, J. (2008). Voice biometrics. In *Handbook of biometrics* (pp. 151-170). Springer US.

[3] Larcher, A., Lee, K. A., Ma, B., & Li, H. (2014). Text-dependent speaker verification: Classifiers, databases and RSR2015. *Speech Communication*, 60,56-77.

[4] D.Ramos Castro, Transparencias de la asignatura: Tratamiento de Señales de Voz y Audio: Introducción a la evaluación de sistemas de reconocimiento, pp.16-21, 2016.

[5] J.Ortega García, Transparencias de la asignatura: Tratamiento de Señales de Voz y Audio: Sistema Auditivo, Sensación Sonora y Parametrización Perceptual, pp.53-58,2016.

[6] The HTK Book (for HTK Version 3.2.1). http://htk.eng.cam.ac.uk/docs/docs.shtml (consultado el 13/05/2016).

[7] Rabiner, L. R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, *77*(2), 257-286.

[8] Documentación online de Kaldi, http://kaldi-asr.org/doc/ (consultado el 18/05/2016).

[9] D.Torre Toledano, J. González Rodríguez, "Transparencias de la asignatura: Tratamiento de Señales de Voz y Audio: HMMs", pp.32-39,2016.

[10] Higgins, A., J. Porter and L. Bahler. YOHO Speaker Authentication Final Report. ITT Defense Communications Division, 1989.

[11] Especificaciones de la base de datos RSR2015, https://www.etpl.sg/innovation-offerings/ready-to-sign-licenses/rsr2015-overview-n-specifications (consultado el 24/05/2016).

[12] Documentación de la Base de datos TIMIT, https://catalog.ldc.upenn.edu/LDC93S1 (consultado el 20/05/2016).

[13] C. Esteve Elizalde, Reconocimiento de locutor dependiente de texto mediante adaptación de modelos ocultos de Markov Fonéticos, Proyecto Fin de Carrera, Escuela Politécnica Superior, UAM, 2007.

Glosario

HMM: Hidden Markov Model.

GMM: Gaussian Mixture Model.

MLLR: Maximum Likelihood Linear Regression.

MAP: Maximum a posteriori.

fMLLR: feature-space Maximum Likelihood Linear Regression.

MFCC: Mel Frequency Cepstral Coefficients.

CMVN: Cepstral Mean and Variance Normalization.

EM: Expectation-Maximization.

PIN: Personal Identification Number.

EER: Equal Error Rate (tasa de igual error).

HTK: Hidden Markov Model Toolkit.

Anexos

A Resultados de verificación de locutor con 18 usuarios

Como ya comentamos en la sección *5.2.2.1*, se realizaron pruebas de verificación de locutor para 18 locutores distintos, con el fin de averiguar si el sistema funcionaba adecuadamente en alguno de ellos. A continuación se presentan las curvas DET obtenidas para los 18 usuarios en cuestión.

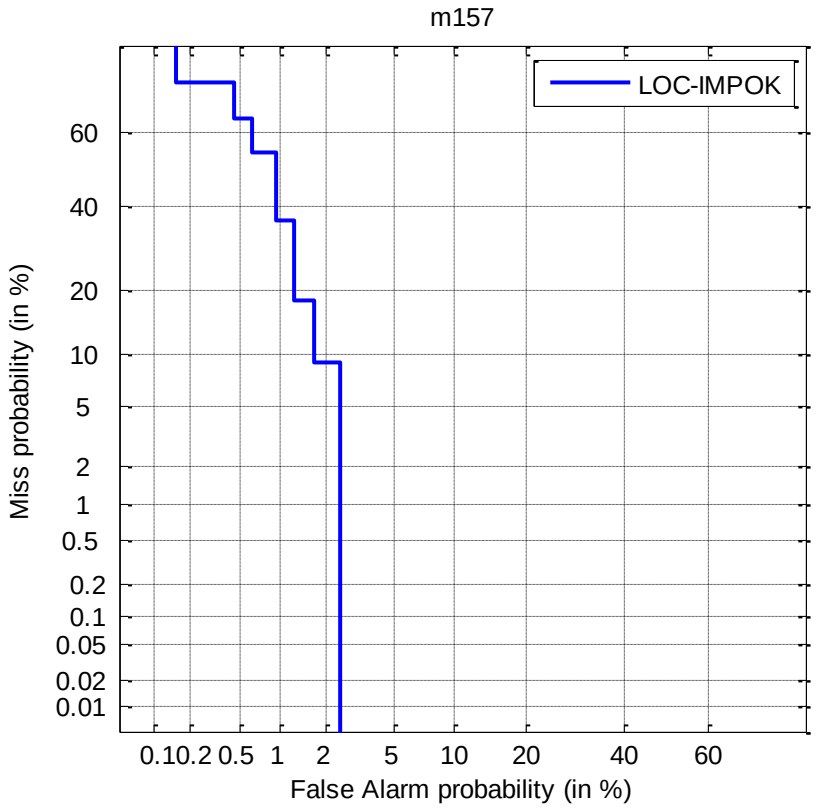

Figura 0-1: Resultados de verificación de locutor para el usuario m157

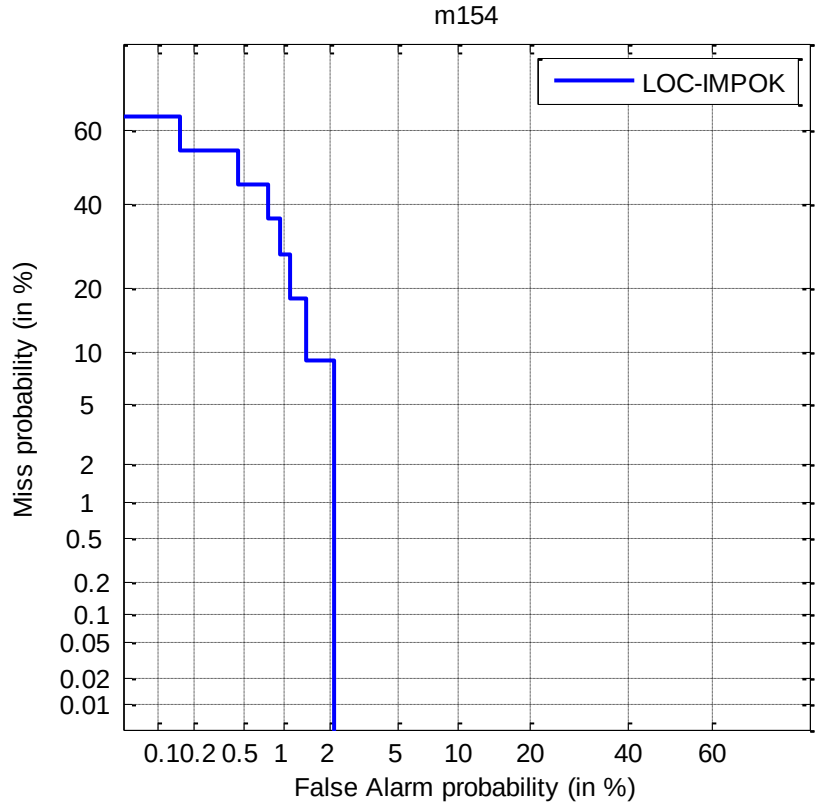

Figura 0-2: Resultados de verificación de locutor para el usuario m154

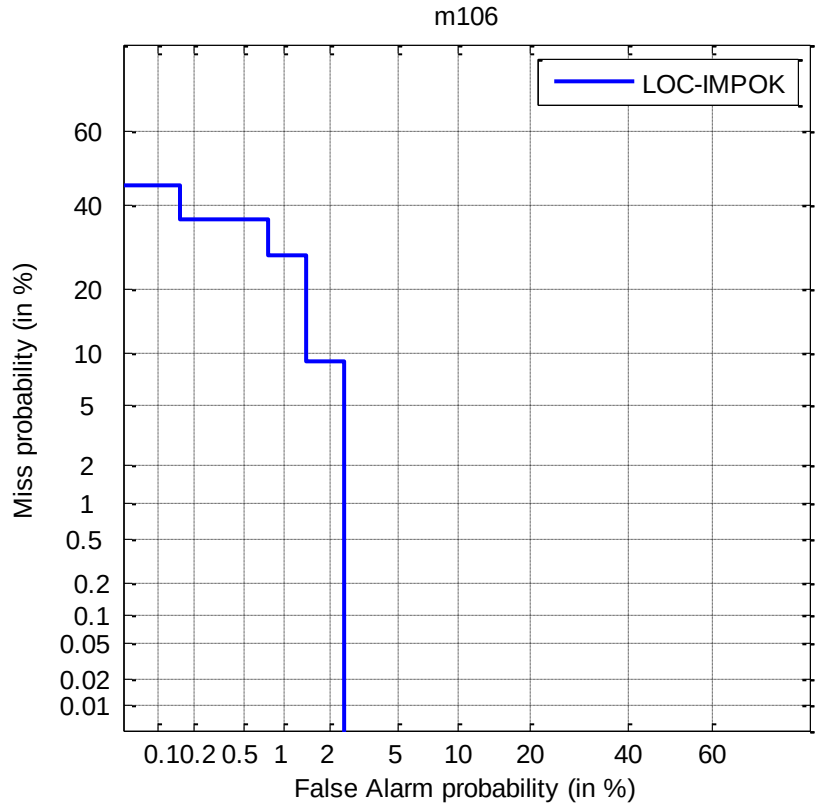

Figura 0-3: Resultados de verificación de locutor para el usuario m106

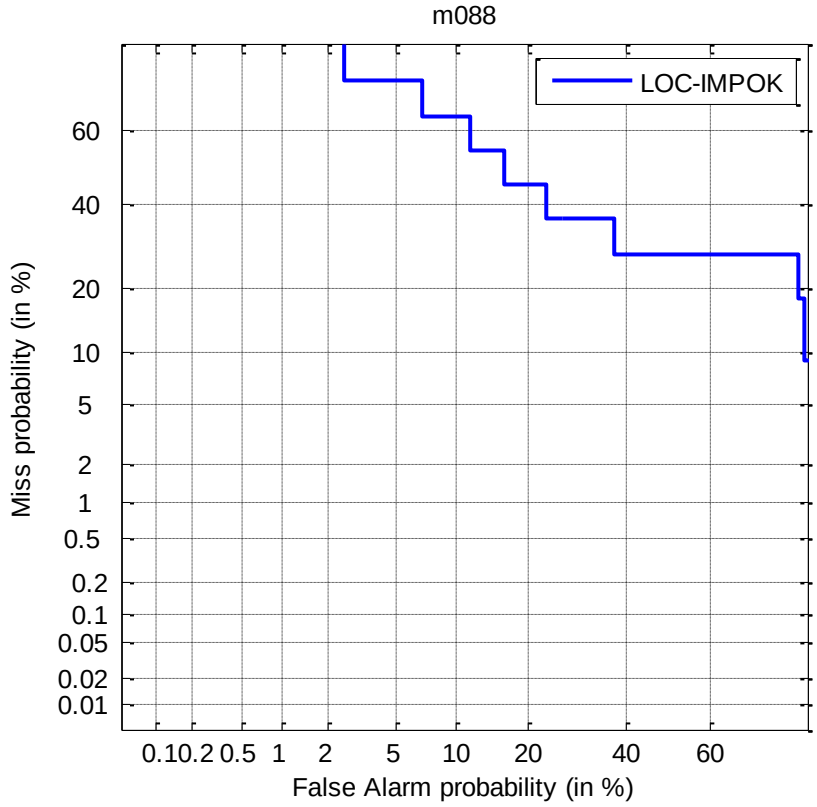

Figura 0-4: Resultados de verificación de locutor para el usuario m088

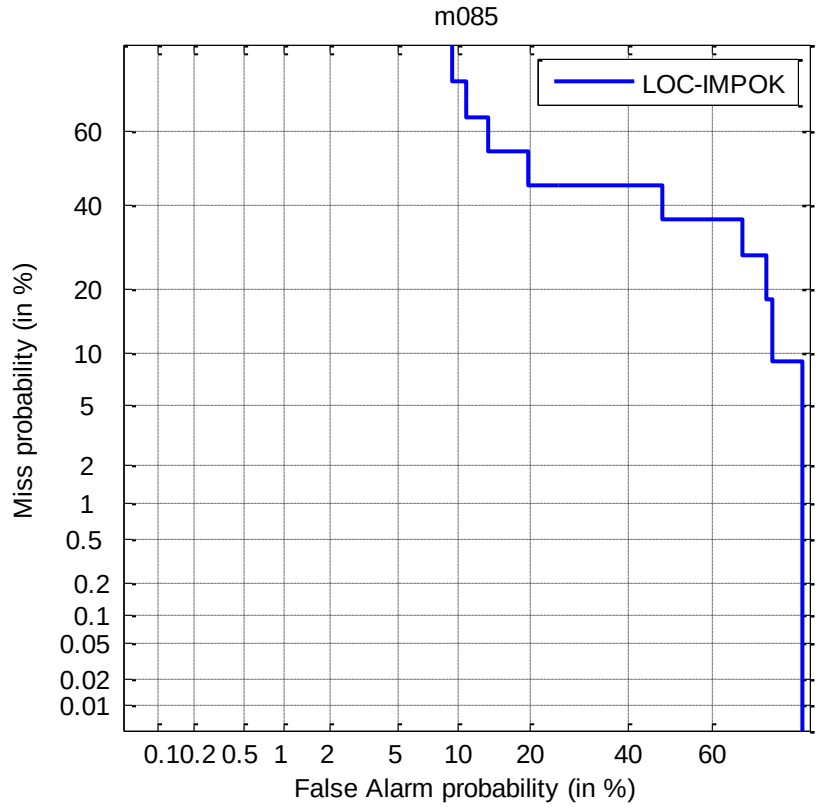

Figura 0-5: Resultados de verificación de locutor para el usuario m085

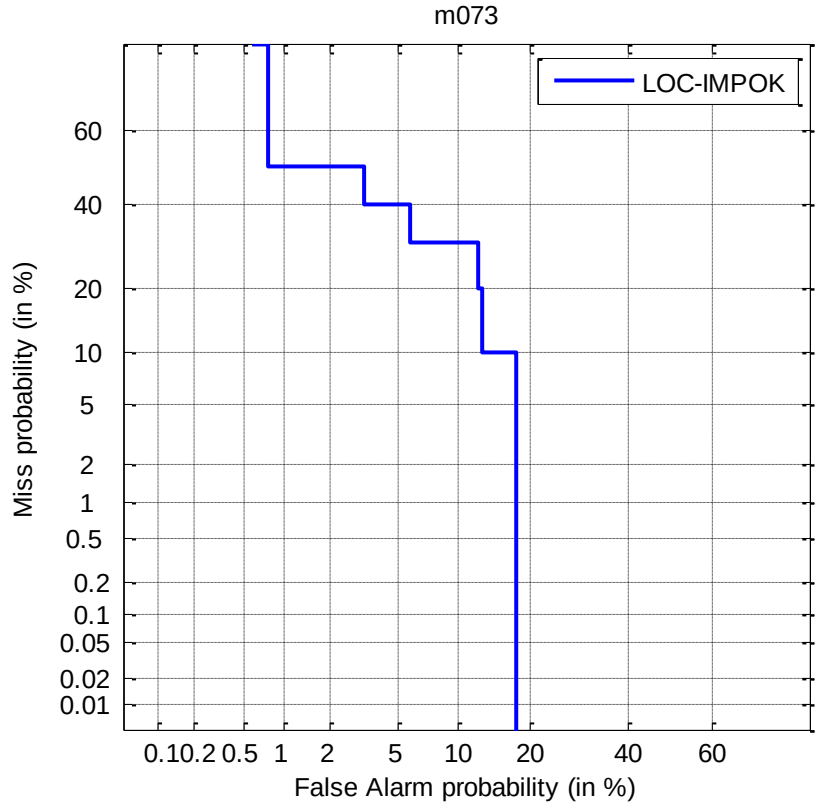

Figura 0-6: Resultados de verificación de locutor para el usuario m073

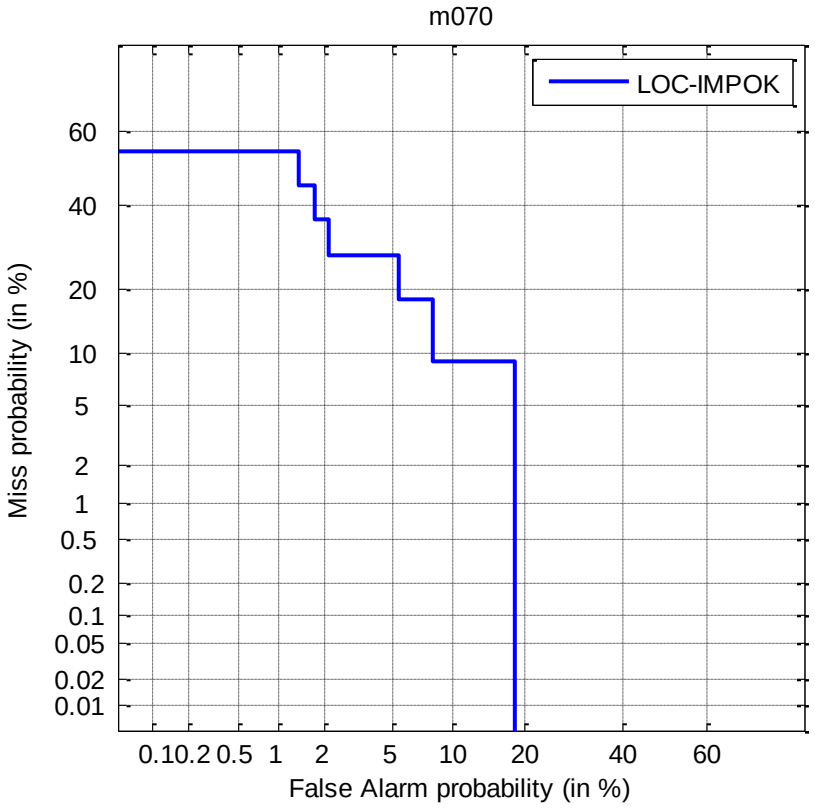

Figura 0-7: Resultados de verificación de locutor para el usuario m070

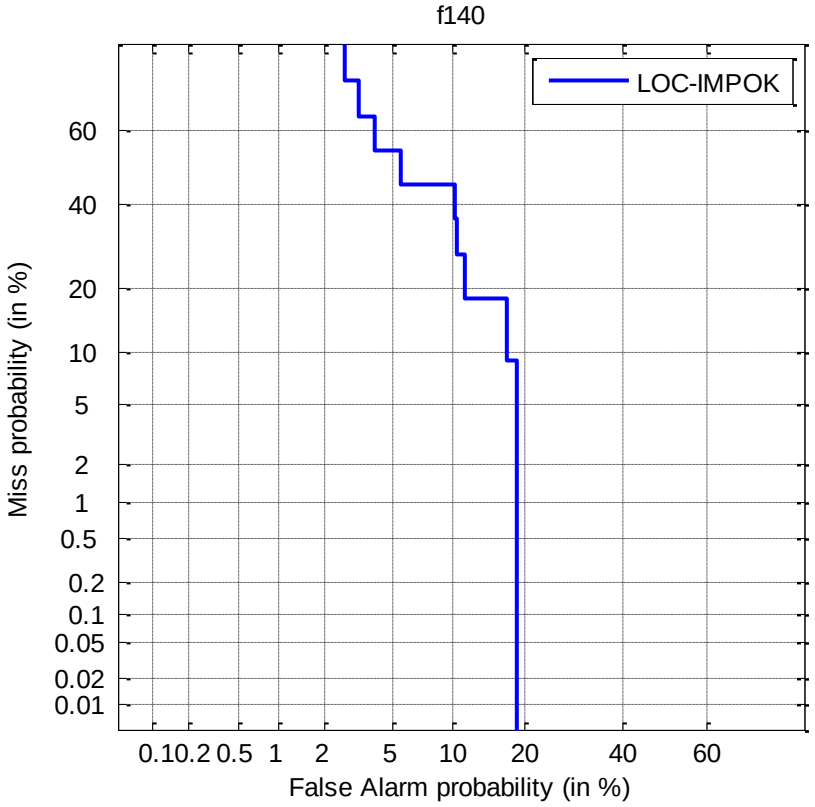

Figura 0-8: Resultados de verificación de locutor para el usuario f140

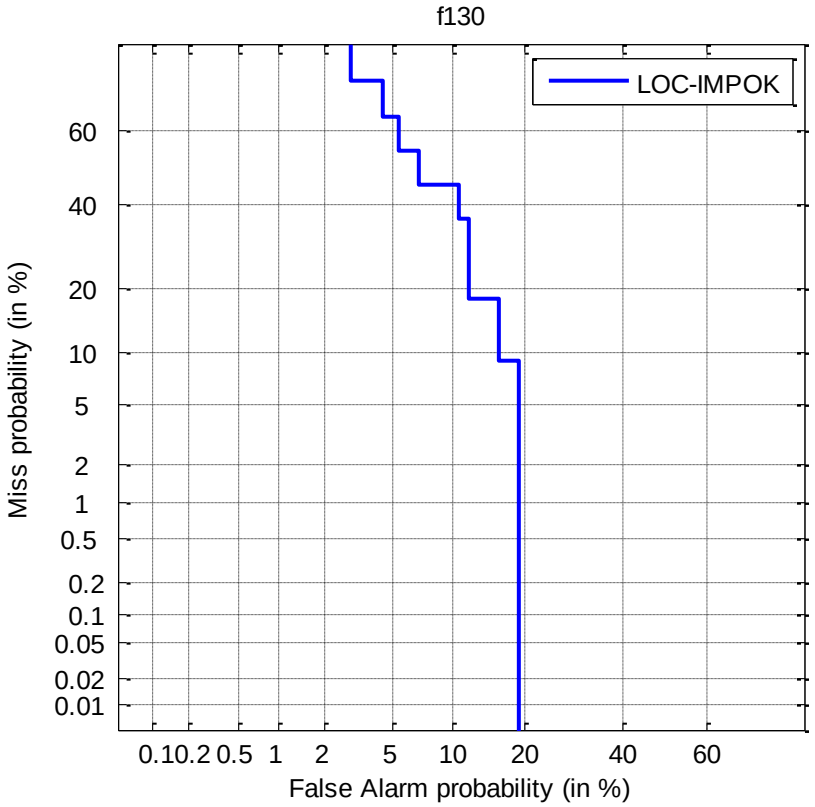

Figura 0-9: Resultados de verificación de locutor para el usuario f130

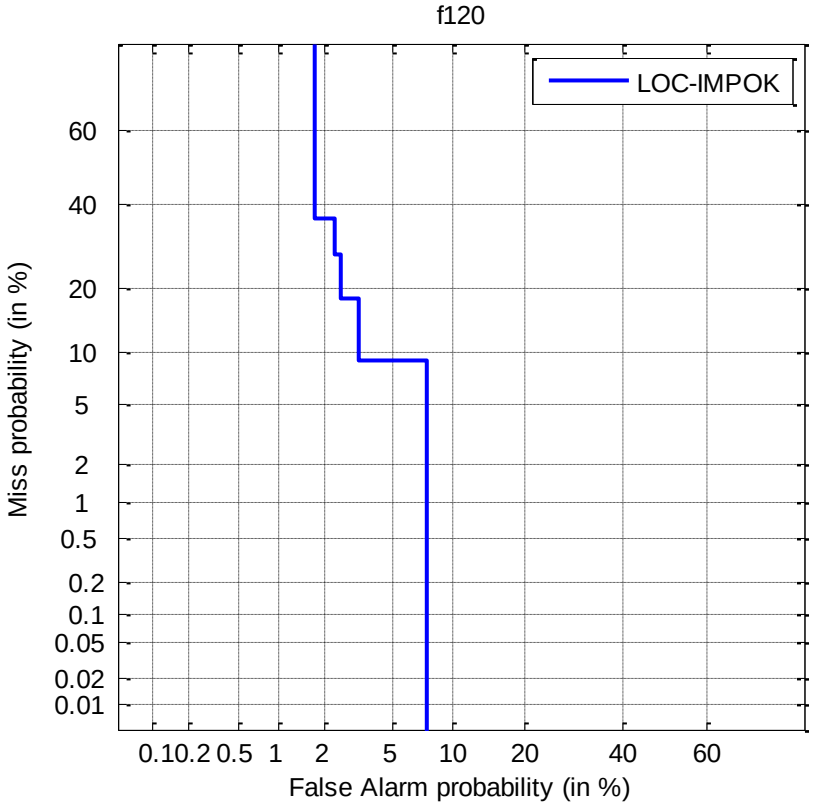

Figura 0-10: Resultados de verificación de locutor para el usuario f120

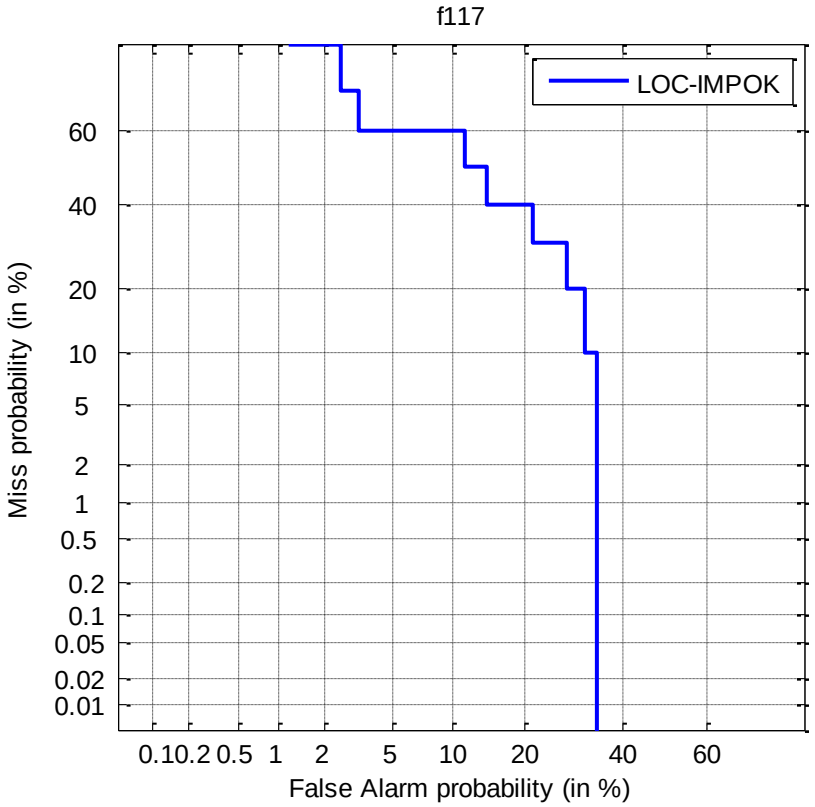

Figura 0-11: Resultados de verificación de locutor para el usuario f117

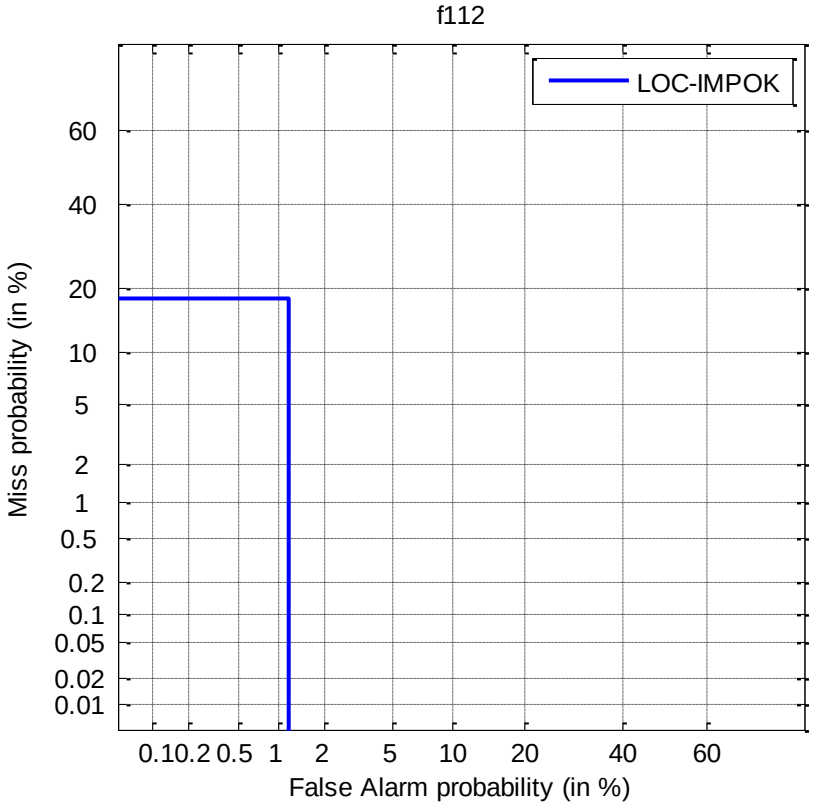

Figura 0-12: Resultados de verificación de locutor para el usuario f112

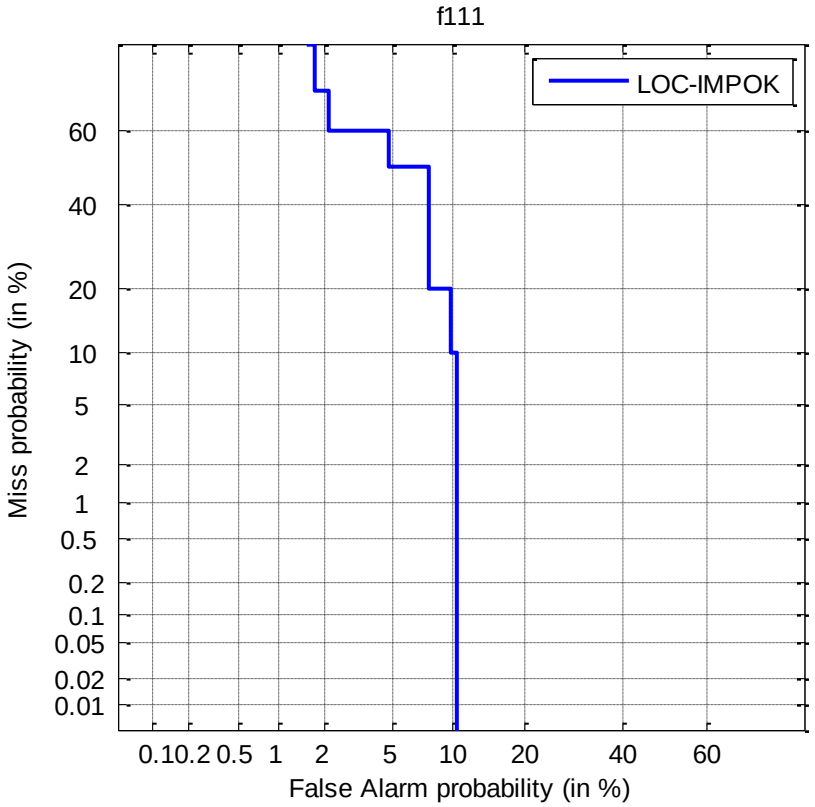

Figura 0-13: Resultados de verificación de locutor para el usuario f111

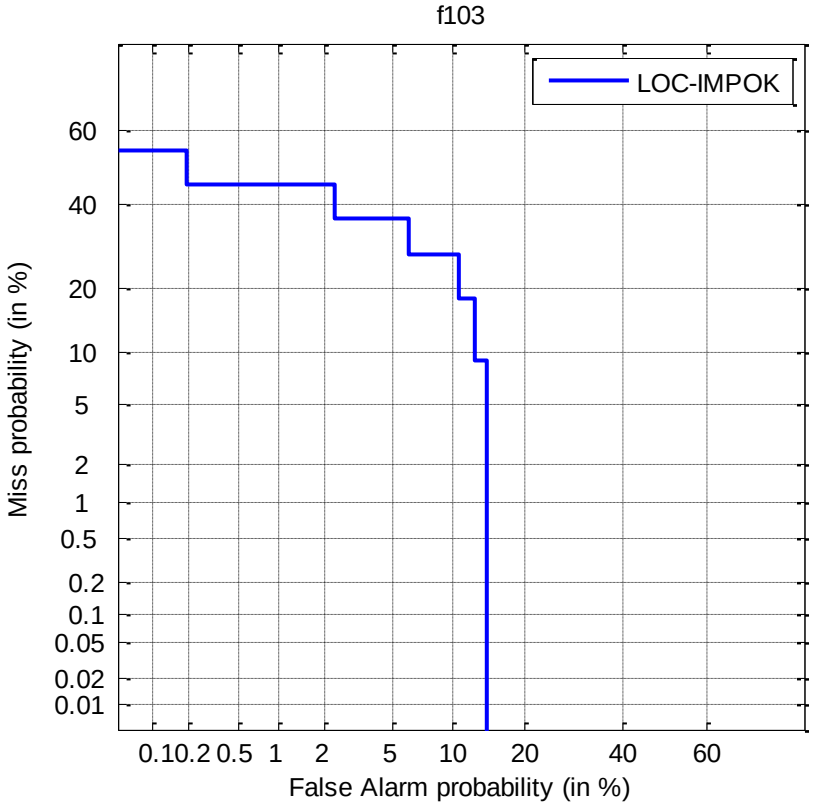

Figura 0-14: Resultados de verificación de locutor para el usuario f103

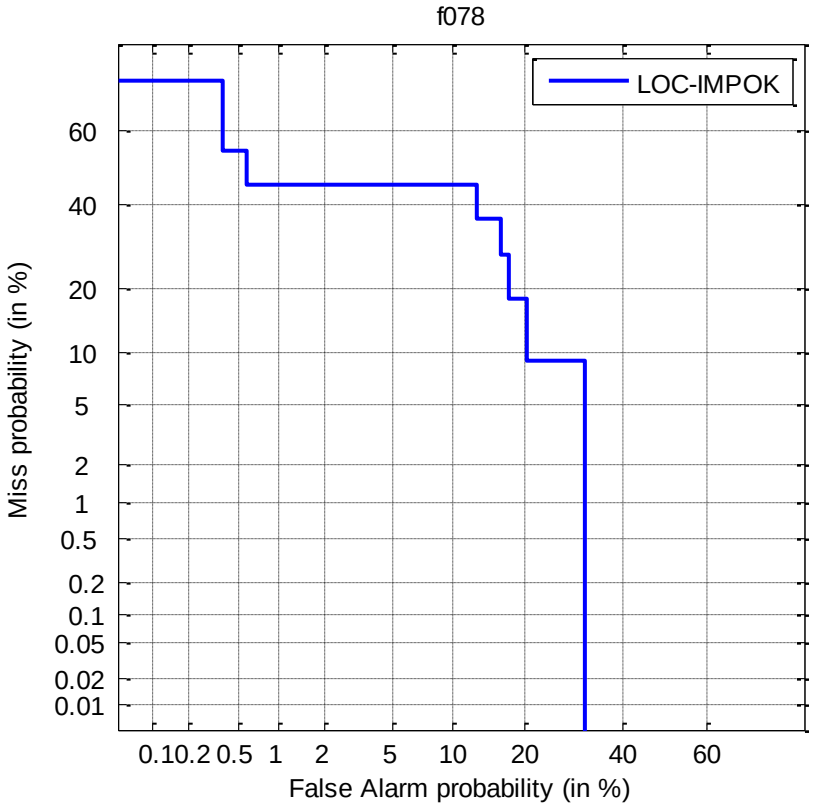

Figura 0-15: Resultados de verificación de locutor para el usuario f078

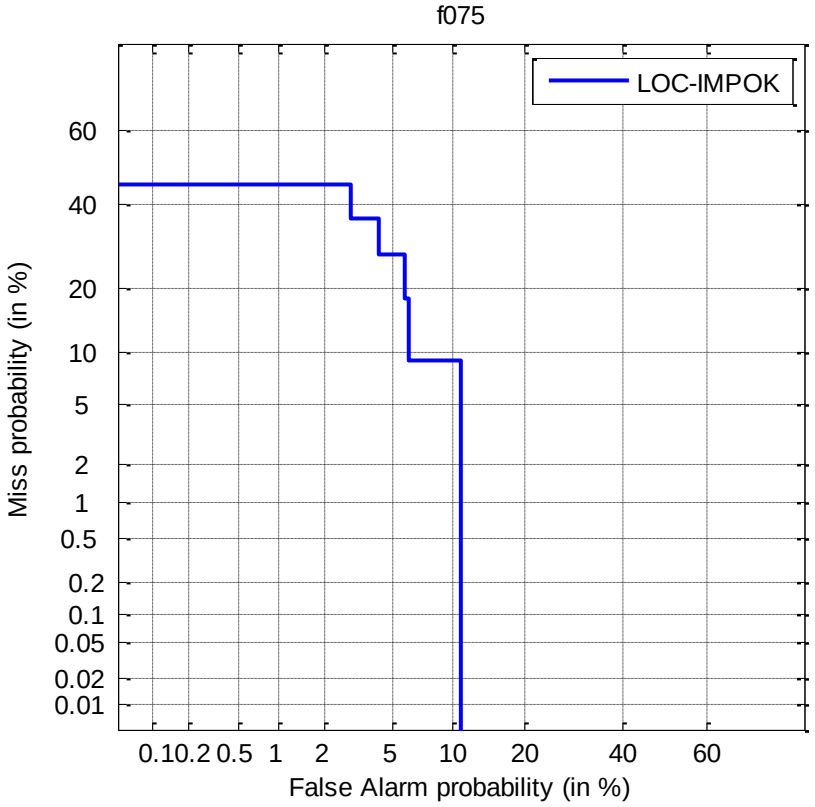

Figura 0-16: Resultados de verificación de locutor para el usuario m075

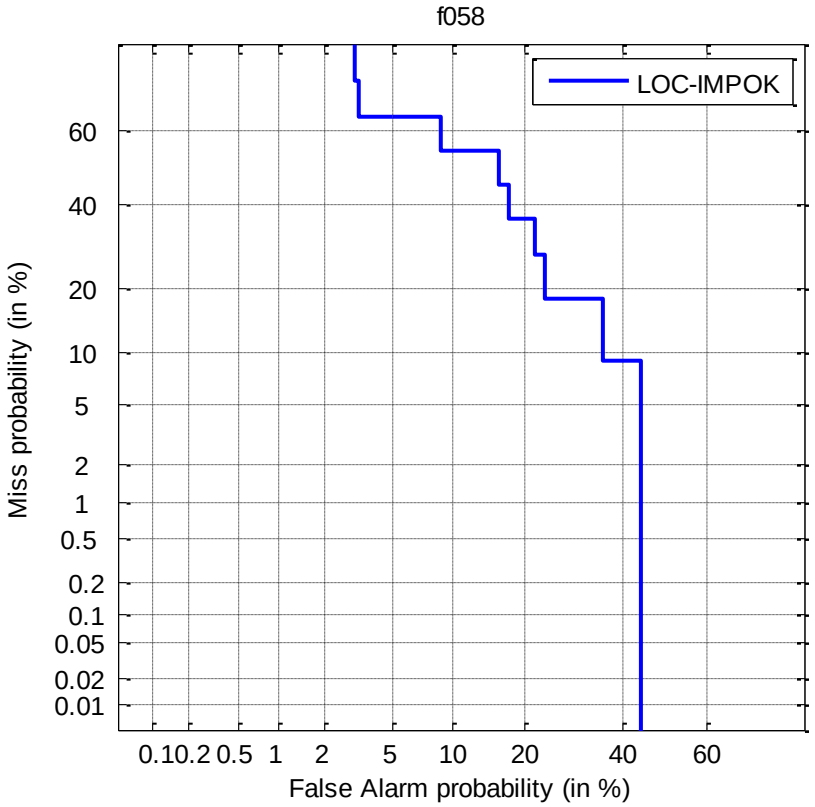

Figura 0-17: Resultados de verificación de locutor para el usuario f058

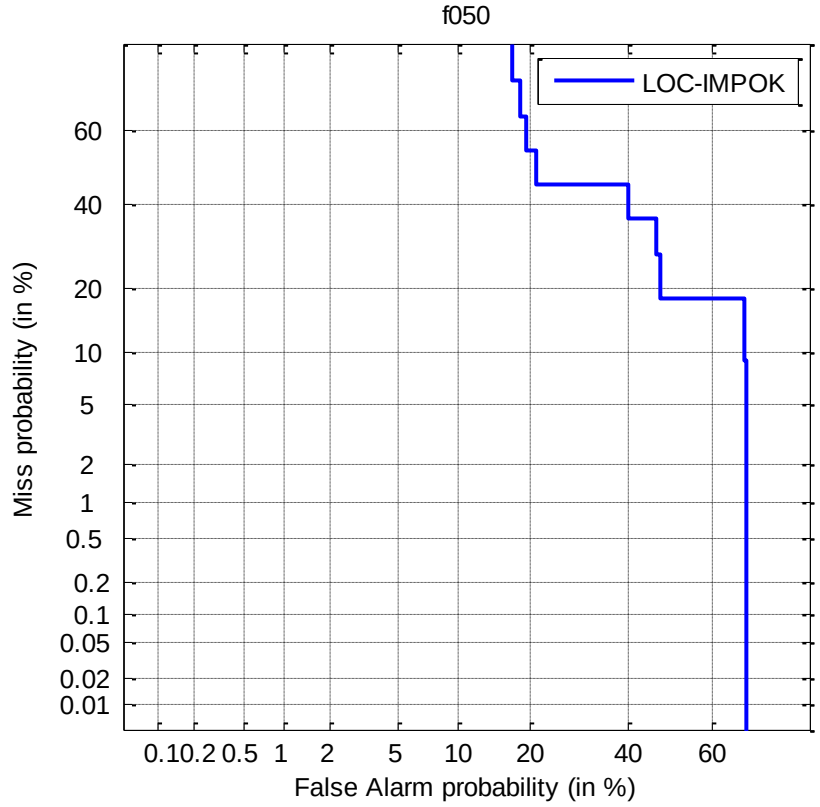

Figura 0-18: Resultados de verificación de locutor para el usuario f050

B Resultados de verificación de frase para distintos N

Como ya se comentó en la sección *5.2.2.2* se analizaron distintos casos para la tarea de verificación de frase. Entre las *N* posibles hipótesis se probó adicionalmente con *N=10, 15, 25,50 y 75.* Los resultados obtenidos fueron los siguientes:

10-best

#matches	1	2	3	4	5	6	7	8	9	10
FA(%)	97.58	76.92	34.77	10.38	1.9	0.17	0	0	0	0
FR(%)	0.24	1	2.13	3.11	4.26	5.69	7.34	9.47	12.91	23.70

Tabla 0-1: Resultados de verificación de frase para 10-best

15-best

#matches	1	2	3	4	5	6	7	8	9	10
FA(%)	97.59	87.78	52.49	18.45	4.39	0.64	0.03	0	0	0
FR(%)	0.22	0.62	1.35	2.09	2.76	4.19	5.89	8.36	12.31	23.47

Tabla 0-2: Resultados de verificación de frase para 15-best

25-best

#matches	1	2	3	4	5	6	7	8	9	10
FA(%)	97.59	89.80	60.48	23.65	6.16	1.09	0.07	0	0	0
FR(%)	0.20	0.51	1.06	1.64	2.24	3.35	5.06	7.49	11.54	23.05

Tabla 0-3: Resultados de verificación de frase para 25-best

50-best

#matches	1	2	3	4	5	6	7	8	9	10
FA(%)	97.11	89.99	65.06	28.28	7.89	1.53	0.11	0	0	0
FR(%)	0.61	0.90	1.24	1.79	2.37	3.27	4.76	7.16	11.28	22.79

Tabla 0-4: Resultados de verificación de frase para 50-best

75-best

#matches	1	2	3	4	5	6	7	8	9	10
FA(%)	97.59	90.46	66.78	30.42	8.80	1.76	0.13	0	0	0
FR(%)	0.19	0.45	0.74	1.19	1.72	2.55	4	6.37	10.36	21.85

Tabla 0-5: Resultados de verificación de frase para 75-best